

AD A101538

AFWAL-TR-81-3034

LEVEL



HYPersonic VEHICLE TRAJECTORY OPTIMIZATION

UNIVERSITY OF TEXAS AT AUSTIN
DEPARTMENT OF AEROSPACE ENGR/ENGR MECHANICS
AUSTIN, TEXAS 78712

May 1981

Final Report for Period January 1980 - December 1980

Approved for public release; distribution unlimited

FLIGHT DYNAMICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

JUL 6 1981

A

417 14 010

NOTICE

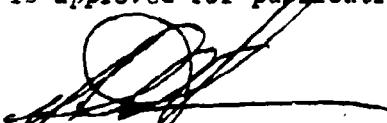
When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

L. Earl Miller

L. Earl Miller,
Project Engineer


Richard D. Neumann
Actg. Chief, High Speed Aero Perf. Br.
Aeromechanics Division

FOR THE COMMANDER


Peter J. Butkewicz, Colonel USAF
Chief, Aeromechanics Division
AF Wright Aeronautical Laboratories (AFSC)

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/PIMG, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

19 REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 18 AFWAL-TR-81-3034	2. GOVT ACCESSION NO. AD-A101538	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Hypersonic Vehicle Trajectory Optimization	5. TYPE OF REPORT & PERIOD COVERED Final Report January 1980 - December 1980 PERFORMING ORG. REPORT NUMBER	
6. AUTHOR(s) 16 David G. Hull and Jason L. Speyer	7. CONTRACT OR GRANT NUMBER(s) F33615-79-C-3030	
8. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Aerospace Engr. & Engr. Mechanics University of Texas at Austin Austin, Texas 78712	9. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 11 2404 07 32	
10. CONTROLLING OFFICE NAME AND ADDRESS Flight Dynamics Laboratory (AFWAL/FIMG) AF Wright Aeronautical Laboratories, AFSC Wright-Patterson Air Force Base, Ohio 45433	11. REPORT DATE 11 May 1981	
12. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	13. NUMBER OF PAGES 113	
14. DISTRIBUTION STATEMENT (of this Report)	15. SECURITY CLASS. (of this report) Unclassified	
Approved for public release, distribution unlimited.	16. DECLASSIFICATION/DOWNGRADING SCHEDULE	
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)	18. SUPPLEMENTARY NOTES	
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Trajectory optimization, Reentry, Orbital Plane Change		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The purpose of this task is to assemble a computer code for solving the maximum crossrange and maximum plane change problems for hypersonic vehicles. These problems are formulated as parameter optimization or nonlinear programming problems by replacing each control function by a number of nodal points and linear interpolation. An existing code for solving the nonlinear programming problem with the augmented-Lagrangian method is used to perform the optimization. Derivatives are calculated		

402465

20. Abstract (cont'd.)

numerically by central differences.

The code works well in that both trajectory optimization problems are solved successfully. By changing that part of the code which defines the problem, this code can be used to solve any trajectory optimization problem.

FOREWORD

This report contains a description of a computer code for solving two hypersonic vehicle trajectory optimization problems, that is, maximum cross-range and maximum plane change. The work was performed by David G. Hull and Jason L. Speyer of the Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Austin, Texas, 78712. The program was sponsored by the Air Force Flight Dynamics Laboratory under Contract F33615-79-C-3030 and work unit number 2404 07 32 and was managed by Dr. L. Earl Miller.

A

TABLE OF CONTENTS

<u>Section</u>	<u>Title</u>	<u>Page</u>
I.	INTRODUCTION.....	1
II.	PHYSICAL MODEL.....	2
	2.1 Equations of Motion.....	2
	2.2 Earth.....	3
	2.3 Atmosphere.....	3
	2.4 Aerodynamics.....	6
	2.5 Propulsion.....	8
	2.6 Performance Indices.....	9
	2.7 Prescribed Boundary Conditions.....	10
	2.8 Inequality Constraints.....	11
	2.9 Nondimensional Variables.....	11
III.	FORMULATION OF THE OPTIMIZATION PROBLEM.....	12
	3.1 Optimal Control Problem.....	12
	3.2 Parameter Optimization Problem.....	14
IV.	PARAMETER OPTIMIZATION METHOD.....	19
	4.1 Augmented-Lagrangian Method.....	19
	4.2 Optimization Code.....	20
V.	USER CODE.....	23
	5.1 Main Program.....	23
	5.2 Performance Index, Constraints, and Derivatives.....	23
VI.	SOLUTION OF THE OPTIMIZATION PROBLEMS.....	25
	6.1 Reentry Problem.....	25
	6.2 Plane Change Problem.....	35
VII.	DISCUSSION AND CONCLUSIONS.....	46
	REFERENCES.....	49
APPENDIX A.	DOCUMENTATION ASSOCIATED WITH THE OPTIMIZATION CODE.....	50
APPENDIX B.	LISTING OF THE OPTIMIZATION CODE.....	61
APPENDIX C.	LISTING OF THE USER CODE.....	89
APPENDIX D.	GLOSSARY OF VARIABLES IN USER CODE.....	102

LIST OF ILLUSTRATIONS

<u>Figure</u>		<u>Page</u>
1.	Maximum Crossrange: α and μ versus t	32
2.	Maximum Crossrange: θ , ϕ , and h versus t	33
3.	Maximum Crossrange: V and ψ versus t	34
4.	Maximum Plane Change: α and μ versus t	42
5.	Maximum Plane Change: θ , ϕ , and h versus t	43
6.	Maximum Plane Change: V , ψ , and i versus t	44

LIST OF TABLES

<u>Table</u>		<u>Page</u>
1.	Atmospheric Constants.....	5
2.	Axial Skin-Friction Force Constants.....	7
3.	Axial Pressure Force Constants.....	7
4.	Normal Force Constants.....	8
5.	Nominal Path for Maximum Crossrange.....	26
6.	Maximum Crossrange Trajectory: Cycle 1 Results.....	28
7.	Maximum Crossrange Trajectory: Cycle 2 Results.....	29
8.	Maximum Crossrange Trajectory: Cycle 3 Results.....	30
9.	Maximum Crossrange Trajectory: Cycle 4 Results.....	31
10.	Nominal Path for Maximum Plane Change	36
11.	Maximum Plane Change: Cycle 1 Results	38
12.	Maximum Plane Change: Cycle 2 Results	39
13.	Maximum Plane Change: Cycle 3 Results	40
14.	Maximum Plane Change: Cycle 4 Results	41

LIST OF SYMBOLS

A_1, B_1, C_1	Coefficients in expression for $C_{A_{SF}}$
A_2, B_2, C_2	Coefficients in expression for $C_{A_{PR}}$
A_3, B_3, C_3	Coefficients in expression for C_N
a	Speed of sound (ft/sec)
a_k	Vector of parameters in control approximation
C	Constraint residuals
$C(t, x, u)$	Control variable inequality constraint
C_A	Axial force coefficient
$C_{A_{PR}}$	Axial force coefficient, pressure
$C_{A_{SF}}$	Axial force coefficient, skin friction
C_D	Drag coefficient
C_L	Lift coefficient
C_N	Normal force coefficient
C_n	Coefficient in standard atmosphere
D	Drag (lb)
D	Nondimensional drag $\equiv D/m_0 g_S$
E	Algebraic inequality constraint
e	Time-dependent inequality constraint
F	Performance index
f	Right-hand side of state differential equation
g	Acceleration of gravity (ft/sec ²)
h	Altitude (ft)
i	Orbital inclination (deg)
J	Performance index
K	Number of inequality constraints

k	Ratio of specific heats
L	Lift (lb)
\tilde{L}	Nondimensional lift $\equiv L/m_0 g_S$
l_n	Temperature gradient in n^{th} layer of standard atmosphere ($^{\circ}\text{K}/\text{ft}$)
M	Mach number
m	Mass (slugs)
\tilde{m}	Nondimensional mass $\equiv m/m_0$
N	Number of unknown parameters
n	Denotes layer of standard atmosphere
R	Gas constant for air ($\text{ft/sec}^2 \text{ deg K}$)
r	Radial distance from center of earth to vehicle (ft)
\tilde{r}	Nondimensional radial distance $\equiv r/r_S$
S	Aerodynamic reference area (ft^2)
$s(t, x)$	State variable inequality constraint
T	Thrust (lb)
\tilde{T}	Nondimensional thrust $\equiv T/m_0 g_S$
t	Time (sec)
\tilde{t}	Nondimensional time $\equiv t/(r_S/g_S)^{1/2}$
t_b	Ignition time
u	Vector of control variables
v	Velocity (ft/sec)
\tilde{v}	Nondimensional velocity $\equiv v/(g_S r_S)^{1/2}$
X	Vector of unknown parameters
x	Vector of state variables
y	Vector of state variables including inequality constraints
α	Angle of attack (deg)
$\tilde{\alpha}$	Angle of attack (rad)

β	Mass flow rate (slugs)
$\tilde{\beta}$	Nondimensional mass flow rate $\equiv (\dot{m}/m_0)(r_s/g_s)^{1/2}$
γ	Flight path inclination (deg)
$\tilde{\gamma}$	Nondimensional flight path inclination (rad)
Δt_b	Burn time
$\tilde{\Delta t}_b$	Nondimensional burn time $\equiv \Delta t_b / (r_s/g_s)^{1/2}$
Θ	Inequality constraints
θ	Longitude (deg)
$\tilde{\theta}$	Nondimensional longitude
θ_i	Parameters in augmented Lagrangian method
μ	Bank angle (deg)
$\tilde{\mu}$	Nondimensional bank angle (rad)
ν	Lagrange multiplier
ρ	Density (slug/ft ³)
σ	Density ratio $\equiv \rho/\rho_s$
σ_i	Weights in augmented Lagrangian method
τ	Absolute temperature (°K)
τ	Normalized time $\equiv t/t_f$
Φ	Performance index in optimal control problem
ϕ	Latitude (deg)
$\tilde{\phi}$	Nondimensional latitude (rad)
Ψ	Vector of constraints in optimal control problem
ψ	Heading angle (deg)
$\tilde{\psi}$	Nondimensional heading angle (rad)
ω	Angular velocity of earth (rad/sec)
$\tilde{\omega}$	Nondimensional angular velocity of earth $\equiv \omega / (r_s/g_s)^{1/2}$

Subscripts

A	Axial
b	Burn
bo	Burnout
D	Drag
f	Final
L	Lift
N	Normal
n	Layer
0	Initial
S	Sea level

Miscellaneous

(')	Derivative with respect to time
()'	Derivative with respect to normalized time
(~)	Nondimensional variable

SECTION I
INTRODUCTION

The purpose of this study is to certify the ability of a particular optimization code to solve trajectory optimization problems associated with hypervelocity vehicles, that is, the maximum crossrange problem and the maximum plane change problem. The optimization problems are converted into parameter optimization problems (nonlinear programming problems) and are solved by the augmented-Lagrangian method. The particular optimization code which will be used has been created at the Atomic Energy Research Establishment in Harwell, England.

The physical model used for the trajectory problems is described in Section II. In Section III, the optimal control problem is converted into a parameter optimization problem, and the optimization code is discussed in Section IV. That part of the code to be provided by the user is presented in Section V. The solution of the optimization problem is carried out in Section VI, and the conclusions are presented in Section VII. Finally, the documentation provided with the optimization code, the listing of the optimization code and user code, and the definition of the variables in the user code is presented in the appendices.

SECTION II

PHYSICAL MODEL

In this section, the physical model used for the reentry and plane change problems is discussed. The model includes the equations of motion, the earth, the atmosphere, the aerodynamics, the propulsion, the performance indices, the boundary conditions, and the inequality constraints.

2.1 Equations of Motion

The equations of motion used for the reentry and plane change problems are those for thrusting flight over a rotating earth. If the vehicle is assumed to be flying west to east and if a positive bank angle is assumed to generate a heading toward the north, these equations are given by

$$\begin{aligned}\dot{\theta} &= V \cos \gamma \cos \psi / r \cos \phi \\ \dot{\phi} &= V \cos \gamma \sin \psi / r \\ \dot{r} &= V \sin \gamma \\ \dot{V} &= (T \cos \alpha - D) / m - g \sin \gamma + \omega^2 r \cos \phi (\sin \gamma \cos \phi - \cos \gamma \sin \phi \sin \psi) \\ \dot{\gamma} &= (T \sin \alpha + L) \cos \mu / mV + (V^2 / r - g) \cos \gamma / V + 2\omega \cos \phi \cos \psi \\ &\quad + (\omega^2 r / V) \cos \phi (\cos \gamma \cos \phi + \sin \gamma \sin \phi \cos \psi) \\ \dot{\psi} &= (T \sin \alpha + L) \sin \mu / mV \cos \gamma - (V / r) \cos \gamma \cos \psi \tan \phi \\ &\quad + 2\omega (\tan \gamma \cos \phi \sin \psi - \sin \phi) - (\omega^2 r / V \cos \gamma) \sin \phi \cos \phi \cos \psi\end{aligned}\tag{1}$$

In these equations, θ is the longitude, ϕ is the latitude, r is the radial distance from the center of the earth to the vehicle center of gravity, V is

the velocity relative to the earth, γ is the flight path inclination, ψ is the heading angle, m is the mass, T is the thrust, D is the drag, L is the lift, α is the angle of attack, μ is the bank angle, and ω is the angular velocity of the earth.

2.2 Earth

The earth is assumed to be a sphere whose radius represents mean sea level and is denoted by r_S . As a consequence, the acceleration of gravity is given by the inverse-square law

$$g = g_S (r_S/r)^2 \quad (2)$$

where g_S is the acceleration of gravity at sea level. Also the altitude of the vehicle above mean sea level satisfies the relation

$$h = r - r_S \quad (3)$$

The values of the constants associated with the characteristics of the earth are listed below:

$$r_S = 20926428 \text{ ft}$$

$$g_S = 32.174 \text{ ft/sec}^2 \quad (4)$$

$$\omega = 7.2921151E-5 \text{ rad/sec}$$

2.3 Atmosphere

In order to obtain atmospheric properties as a function of altitude, the approximate atmosphere known as the 1962 U.S. Standard Atmosphere has been employed, with the additional assumption that the composition of the atmosphere

is constant. Here, the atmosphere is divided into a number of layers in which the temperature gradient is assumed constant. Hence, the absolute temperature in each layer is given by

$$\tau = \tau_n + l_n(h - h_n), \quad h \geq h_n \quad (5)$$

where τ_n is the absolute temperature at the beginning of the n^{th} layer, l_n is the constant temperature gradient, and h_n is the altitude at the beginning of the layer. Once the temperature is known, the speed of sound is obtained from the relation

$$a = (kR\tau)^{1/2} \quad (6)$$

where k is the ratio of specific heats of air and R is the gas constant of air at sea level. For those layers where the temperature gradient is nonzero, the density ratio, $\sigma = \rho/\rho_S$, can be expressed as

$$\sigma = C_n \tau_n^{-(1 + g_S/R l_n)} \quad (7)$$

where ρ_S is the density at sea level and C_n is a known constant for each layer. For those layers where the temperature gradient is zero, the density ratio becomes

$$\sigma = C_n \exp(-g_S h_n/R \tau_n) \quad (8)$$

The values of the constants associated with the atmosphere are presented below and in Table 1:

$$k = 1.4$$

$$R = 3086.9629 \text{ ft}^2/\text{sec}^2 \text{ }^{\circ}\text{K}$$

$$\rho_S = 2.3769E-3 \text{ slugs/ft}^3 \quad (9)$$

Table 1. Atmospheric Constants

Layer (n)	h_n (ft)	τ_n (°K)	l_n (°K/ft)	c_n
1	0	288.15	-1.9812E-3	3.401824655257E-11
2	36,089	216.65	0	1.683376997149E+00
3	65,617	216.65	3.0480E-4	9.817858914969E+80
4	104,987	228.65	8.5344E-4	1.506414967722E+29
5	154,199	270.65	0	4.394749884481E-01
6	170,604	270.65	-6.0960E-4	4.717851690435E-43
7	200,131	252.65	-1.2192E-3	1.562793740651E-22
8	259,186	180.65	0	5.028946984109E+01

During the optimization process, trajectories can be obtained which go to very high or very low altitudes. To avoid a fatal error associated with exponential overflow, the following additional features have been incorporated in the model atmosphere:

$$\begin{aligned}
 h \leq 0 : \quad \tau &= 288.15 - 1.9812E-3 h \\
 \sigma &= 1. - 2.9262913E-5 h \\
 h \geq 750,000 : \quad \tau &= 180.65 \\
 \sigma &= 8.111816E-18
 \end{aligned} \tag{10}$$

Hence, below sea level, the density ratio is assumed to vary linearly with altitude, and at high altitudes, it is assumed constant at the value for 750,000 ft.

2.4 Aerodynamics

The drag and the lift are related to the drag coefficient C_D and the lift coefficient C_L as follows:

$$D = (1/2) \rho V^2 C_D \quad (11)$$

$$L = (1/2) \rho V^2 C_L$$

where S is the aerodynamic reference area. The drag and lift coefficients can be expressed in terms of the axial and normal force of coefficients as

$$C_D = C_A \cos \alpha + C_N \sin \alpha \quad (12)$$

$$C_L = C_N \cos \alpha - C_A \sin \alpha$$

The axial force coefficient is composed of a skin-friction term and a pressure term, that is,

$$C_A = C_{A_{SF}} + C_{A_{PR}} \quad (13)$$

where

$$C_{A_{SF}} = A_1 h^2 + B_1 h + C_1 \quad (14)$$

$$C_{A_{PR}} = A_2 \alpha^2 + B_2 \alpha + C_2$$

In these relations, A_1 , B_1 , and C_1 are known functions of the Mach number, $M = V/a$, while A_2 , B_2 and C_2 are known functions of Mach number and angle of attack. Finally, the normal force coefficient is given by

$$C_N = A_3\alpha^2 + B_3\alpha + C_3 \quad (15)$$

where A_3 , B_3 , and C_3 are known functions of Mach number.

The values of the constants associated with the aerodynamics are given below and in Tables 2 through 4.

$$S = 125.84 \text{ ft}^2 \quad (16)$$

Table 2. Axial Skin-Friction Force Constants

M	A_1	B_1	C_1
0.2	0	7.80E-8	.0114
1.2	0	7.70E-8	.0076
5.0	0	5.60E-8	.0028
10.	2.40E-12	-7.11E-7	.0578
$\geq 20.$	1.38E-12	-3.81E-7	.0297

Table 3. Axial Pressure Force Constants

M	A_2		B_2		C_2	
	$\alpha < 16^\circ$	$\alpha \geq 16^\circ$	$\alpha < 16^\circ$	$\alpha \geq 16^\circ$	$\alpha < 16^\circ$	$\alpha \geq 16^\circ$
0.2	-1.00E-4	-1.00E-4	-2.19E-3	-2.19E-3	.0350	.0350
1.2	-7.81E-5	-7.81E-5	-1.25E-3	-1.25E-3	.0760	.0760
5.0	1.09E-5	4.86E-6	-9.62E-4	-1.13E-4	.0324	.0204
10.	1.41E-5	-6.60E-6	-9.00E-4	3.49E-4	.0261	.0114
$\geq 20.$	1.33E-5	-6.25E-6	-8.19E-4	3.58E-4	.0243	.0105

Table 4. Normal Force Constants

M	A ₃	B ₃	C ₃
0.2	1.33E-4	2.68E-2	-.030
1.2	-7.81E-5	2.90E-2	-.030
5.0	2.94E-4	1.41E-2	-.035
10.	4.38E-4	6.50E-2	-.035
<u>> 20.</u>	4.38E-4	6.50E-2	-.035

In the computer program, the above tables are read by linear interpolation. Also, for $M \geq 20$, the $M = 20$ value is used because the extrapolation of the above numbers could lead to wrong results.

2.5 Propulsion

For the plane change problem, rocket engines are used to aid in the plane change and to increase the energy of the vehicle. It is assumed that the engines burn once and that the propellant mass flow rate and the thrust are constant during the burn. Hence, during the burn, the mass of the vehicle satisfies the relation

$$m = m_b - \beta(t - t_b) \quad (17)$$

where β is the mass flow rate and where m_b and t_b are the mass and the time at the initiation of the burn.

The values of the constants associated with the propulsion characteristics are as follows:

$$m_b = m_0 = 335.67477 \text{ slugs}$$

$$\beta = 0.34768573 \text{ slugs/sec}$$

$$T = 3300 \text{ lb} \quad (18)$$

$$m_p = 179.18195 \text{ slugs}$$

$$\Delta t_b = 515.35606 \text{ sec}$$

where m_p is the propellant mass and Δt_b is the total burn time. In the reentry program, the propellant mass is assumed to have been expended, so the initial mass is $m_0 = 156.49282$ slugs.

2.6 Performance Indices

In the reentry problem, it is desired to maximize the crossrange. Hence, to have a minimization problem, the performance index is written as

$$J = -\phi_f \quad (19)$$

For the orbital plane change problem, it is desired to maximize the plane change. Maximizing the plane change is equivalent to maximizing the orbit inclination because the initial orbit inclination is zero. The orbit inclination is the angle between the inertial angular momentum vector

$$\bar{H} = \bar{r} \times \bar{v}_{\text{inertial}} \quad (20)$$

and the angular velocity vector of the earth, where the bar denotes a vector. From the definition of the dot product, it is seen that

$$\cos i = (\bar{H} \cdot \bar{\omega}) / H\omega \quad (21)$$

Also, the inertial velocity is given by

$$\bar{v}_{\text{inertial}} = \bar{v} + \bar{\omega} \times \bar{r} \quad (22)$$

Finally, if the vector operations are carried out, the problem of maximizing i becomes that of minimizing the cosine of the inclination at the final point:

$$J = (\cos i)_f \quad (23)$$

where

$$\cos i = \frac{\cos \phi (V \cos \gamma \cos \psi + r\omega \cos \phi)}{[(V \cos \gamma \cos \psi + r\omega \cos \phi)^2 + (V \cos \gamma \sin \psi)^2]^{1/2}} \quad (24)$$

2.7 Prescribed Boundary Conditions

The initial conditions for both problems are as follows:

$$t_0 = 0, \quad \theta_0 = 0, \quad \phi_0 = 0, \quad h_0 = 364,800 \quad (25)$$

$$v_0 = 24,500 \text{ ft/sec}, \quad \gamma_0 = -1.2 \text{ deg}, \quad \psi_0 = 0$$

For the reentry problem, the final condition is given by

$$h_f = 100,000 \text{ ft} \quad (26)$$

The final conditions for the plane change problem are the following:

$$h_f = 608,000 \text{ ft}, \quad v_f = 23,500 \text{ ft/sec}, \quad \gamma_f = 0 \quad (27)$$

2.8 Inequality Constraints

The angle of attack and the load factor, $n = L/W$, are required to satisfy the inequality constraints

$$\alpha \leq 40 \text{ deg} , \quad n \leq 4.5 \quad (28)$$

2.9 Nondimensional Variables

Nondimensionalization has the effect of scaling the variables and can be very beneficial to optimization. In this system, all angles are in radians. Furthermore, to simplify the notation, a nondimensional variable is denoted by the same symbol as the dimensional variable and a tilde. Hence, the following nondimensional variables are defined:

$$\begin{aligned} t/(r_S/g_S)^{1/2} &\equiv \tilde{t} , \quad V/(g_S r_S)^{1/2} \equiv \tilde{V} , \quad r/r_S \equiv \tilde{r} , \quad \omega(r_S/g_S)^{1/2} \equiv \tilde{\omega} \\ T/m_0 g_S &\equiv \tilde{T} , \quad D/m_0 g_S \equiv \tilde{D} , \quad L/m_0 g_S \equiv \tilde{L} , \quad m/m_0 \equiv \tilde{m} , \quad \beta(r_S/g_S)^{1/2}/m_0 \equiv \tilde{\beta} \end{aligned} \quad (29)$$

SECTION III
FORMULATION OF THE OPTIMIZATION PROBLEM

In this section, the optimal control problem is stated in terms of the nondimensional variables, and the parameter optimization problem is formulated.

3.1 Optimal Control Problem

The optimal control problem is to find the angle of attack and roll angle histories which maximize the crossrange for the reentry problem which maximize the orbit inclination (or minimize its cosine) for the plane change problem. In terms of the nondimensional variables presented in Equation (29), the optimization problem can be stated as the following minimization problem:

Find the control variable histories $\tilde{\alpha}(t)$ and $\tilde{\psi}(t)$ which minimize the performance index

$$\begin{aligned} \text{Reentry} \quad J &= -\tilde{\phi}_f \\ \text{Plane Change} \quad J &= \left\{ \frac{\cos \tilde{\phi} (\tilde{V} \cos \tilde{\gamma} \cos \tilde{\psi} + \tilde{r} \tilde{\omega} \cos \tilde{\phi})}{[(\tilde{V} \cos \tilde{\gamma} \cos \tilde{\psi} + \tilde{r} \tilde{\omega} \cos \tilde{\phi})^2 + (\tilde{V} \cos \tilde{\gamma} \sin \tilde{\psi})^2]^{1/2}} \right\}_f \end{aligned} \quad (30)$$

subject to the differential constraints

$$d\tilde{\theta}/d\tilde{t} = \tilde{V} \cos \tilde{\gamma} \cos \tilde{\psi} / (\tilde{r} \cos \tilde{\phi})$$

$$d\tilde{\phi}/d\tilde{t} = \tilde{V} \cos \tilde{\gamma} \sin \tilde{\psi} / \tilde{r}$$

$$d\tilde{r}/d\tilde{t} = \tilde{V} \sin \tilde{\gamma}$$

$$\begin{aligned} d\tilde{V}/d\tilde{t} &= (\tilde{T} \cos \tilde{\alpha} - \tilde{D})/\tilde{m} - \sin \tilde{\gamma} / \tilde{r}^2 + \tilde{\omega}^2 \tilde{r} \cos \tilde{\phi} (\sin \tilde{\gamma} \cos \tilde{\phi} \\ &\quad - \cos \tilde{\gamma} \sin \tilde{\phi} \sin \tilde{\psi}) \end{aligned}$$

$$\begin{aligned}
 \frac{d\tilde{\gamma}}{d\tilde{t}} &= (\tilde{T} \sin \tilde{\alpha} + \tilde{L}) \cos \tilde{u}/\tilde{m}\tilde{V} + \tilde{V} \cos \tilde{\gamma}/\tilde{r} - \cos \tilde{\gamma}/\tilde{r}^2 \tilde{V} + 2\tilde{\omega} \cos \tilde{\phi} \cos \tilde{\psi} \\
 &+ (\tilde{\omega}^2 \tilde{r}/\tilde{V}) \cos \tilde{\phi} (\cos \tilde{\gamma} \cos \tilde{\phi} + \sin \tilde{\gamma} \sin \tilde{\phi} \sin \tilde{\psi}) \\
 (31) \quad &
 \end{aligned}$$

$$\begin{aligned}
 \frac{d\tilde{\psi}}{d\tilde{t}} &= (\tilde{T} \sin \tilde{\alpha} + \tilde{L}) \sin \tilde{u}/\tilde{m}\tilde{V} \cos \tilde{\gamma} - (\tilde{V} \cos \tilde{\gamma}/\tilde{r}) \cos \tilde{\psi} \tan \tilde{\phi} \\
 &+ 2\tilde{\omega} (\tan \tilde{\gamma} \cos \tilde{\phi} \sin \tilde{\psi} - \sin \tilde{\phi}) \\
 &- (\tilde{\omega}^2 \tilde{r}/\tilde{V} \cos \tilde{\gamma}) \sin \tilde{\phi} \cos \tilde{\phi} \cos \tilde{\psi} ,
 \end{aligned}$$

subject to the prescribed boundary conditions

$$\begin{aligned}
 \tilde{t}_0 &= 0, \quad \tilde{\theta}_0 = 0, \quad \tilde{\phi}_0 = 0, \quad \tilde{r}_0 = 1.017432502 \\
 (32) \quad & \\
 \tilde{v}_0 &= 0.94420438, \quad \tilde{\gamma}_0 = -2.0943951E-2, \quad \tilde{\psi}_0 = 0
 \end{aligned}$$

Reentry: $\tilde{r}_f = 1.0047786464$

Plane Change: $\tilde{r}_f = 1.029054170, \quad \tilde{v}_f = 0.90566542, \quad \tilde{\gamma}_f = 0,$

and subject to the inequality constraints

$$\tilde{\alpha} \leq 0.69813170, \quad n \leq 4.5. \quad (33)$$

For the reentry problem, the vehicle is gliding with the engine off. Hence, the thrust, the mass flow rate, and the mass satisfy the relations

$$\tilde{T} = 0, \quad \tilde{\beta} = 0, \quad \tilde{m} = 1 \quad (34)$$

while the drag and lift are given by

$$\begin{aligned}
 \tilde{D} &= 15268.635 C_D \tilde{V}^2 \\
 (35) \quad & \\
 \tilde{L} &= 15268.635 C_L \tilde{V}^2
 \end{aligned}$$

For the plane change problem, the engines are ignited at \tilde{t}_b , whose optimal value is to be determined, and burn for the time period $\Delta\tilde{t}_b = 0.63901697$.

Here, the thrust, the mass flow rate, and the mass are given by

$$\tilde{T} = \begin{cases} 0 \\ 0.30555556 \end{cases} \quad \tilde{\beta} = \begin{cases} 0 & \text{engine off} \\ 0.83533982 & \text{engine on} \end{cases} \quad (36)$$

$$\tilde{m} = \begin{cases} 1. & , \quad 0 \leq \tilde{t} \leq \tilde{t}_b \\ 1. - \tilde{\beta}(\tilde{t} - \tilde{t}_b) & , \quad \tilde{t}_b \leq \tilde{t} \leq \tilde{t}_b + \Delta\tilde{t}_b = \tilde{t}_{bo} \\ 0.46620367 & , \quad \tilde{t}_{bo} \leq \tilde{t} \leq 1 \end{cases}$$

while the drag and the lift become

$$\begin{aligned} \tilde{D} &= 932.34367 C_D \tilde{V}^2 \\ \tilde{L} &= 932.34367 C_L \tilde{V}^2 \end{aligned} \quad (37)$$

The reason for the different constants in Equations (36) and (37) is that the initial mass for the reentry problem is less than the initial mass for the plane change problem (see Section 2.5).

3.2 Parameter Optimization Problem

In general, the optimal control problem of the previous section can be stated in standard matrix notation (Reference 1) as follows:

Mimimize the performance index

$$J = \Phi(t_f, x_f) \quad (38)$$

subject to the differential constraints

$$\dot{x} = f(t, x, u), \quad (39)$$

the prescribed boundary conditions

$$t_0 = 0, \quad x_0 \text{ given}, \quad \psi(t_f, x_f) = 0, \quad (40)$$

the control variable inequality constraints

$$C(t, x, u) \geq 0, \quad (41)$$

and the state variable inequality constraints

$$S(t, x) \geq 0. \quad (42)$$

In the parameter optimization method which will be used to solve these problems, the inequality constraints must be in an algebraic form. This can be accomplished by forming an integral over the region where the inequality constraint is violated. In other words, if $e(t) \geq 0$ is a scalar inequality constraint, an appropriate integral constraint is

$$E = - \int_{t_0}^{t_f} [\min(e, 0)]^2 dt \geq 0 \quad (43)$$

This integral constraint can be converted into a differential equation and an algebraic inequality constraint. Let

$$\dot{x}_{n+1} = -[\min(e, 0)]^2, \quad x_{n+1,0} = 0 \quad (44)$$

so that Equation (43) becomes

$$E = x_{n+1,f} \geq 0 \quad (45)$$

If the inequality constraints are converted into differential equations and algebraic constraints, the optimal control problems can be restated as follows:

Minimize the performance index

$$J = \Phi(t_f, y_f) \quad (46)$$

subject to

$$\dot{y} = f(t, y, u) \quad (47)$$

$$t_0 = 0, \quad y_0 = \text{given},$$

to

$$\Psi(t_f, y_f) = 0, \quad (48)$$

and to

$$\Theta(t_f, y_f) \geq 0. \quad (49)$$

The vector y now contains the original state x and the states introduced by converting the inequality constraints.

The conversion of this optimal control problem into a parameter optimization or nonlinear programming problem is performed in two steps. The first step is to convert the final time into a parameter, and the second step is to parameterize the control histories.

The final time can be made into a parameter by introducing the transformation

$$\tau = t/t_f. \quad (50)$$

This transformation allows the optimal control problem to be rewritten as

Minimize the performance index

$$J = \Phi(y_f, t_f) \quad (51)$$

subject to

$$y' = g(\tau, y, u, t_f) \quad (52)$$

$$\tau_0 = 0, \quad y_0 = \text{given}, \quad \tau_f = 1,$$

to

$$\psi(y_f, t_f) = 0, \quad (53)$$

and to

$$\theta(y_f, t_f) \geq 0. \quad (54)$$

Note that the prime denotes a derivative with respect to τ and that this transformation fixes the interval of integration.

The control histories can be parameterized in a number of ways, but the simplest is to use a set of nodal points and create the function by linear interpolation (see Figure 1 where α and μ are control variables). This approach allows the k^{th} control to be written in the form

$$u_k = u_k(\tau, a_k) \quad (55)$$

where a_k is a vector of parameters, that is, the nodal points defining the control history u_k .

At this point, the unknown parameters are lumped into a single vector X defined as

$$X = [t_f, a_1, \dots, a_p] \quad (56)$$

where p is the number of control functions. For the plane change problem, X also contains the ignition time, t_b , as the last parameter. The total number of parameters is N . If values are given to the components of X , that is, if the final time, the control histories, and the ignition time are known, the differential equations (52) can be integrated from $\tau_0 = 0$ to $\tau_f = 1$ to form the function $y_f = y_f(X)$. In turn, y_f can be eliminated from the performance index and the constraints to form the parameter version of the optimal control problem:

Minimize the performance index

$$J = F(X) \quad (57)$$

subject to the equality constraints

$$C_i(X) = 0, \quad i = 1 \dots K \quad (58)$$

and the inequality constraints

$$C_i(X) \geq 0, \quad i = K + 1 \dots M \quad (59)$$

SECTION IV
PARAMETER OPTIMIZATION METHOD

In this section, the theoretical basis of the method which has been selected for solving the optimization problem is discussed. Also, some remarks are made about the code.

4.1 Augmented-Lagrangian Method

The augmented-Lagrangian method is a particular formulation of the penalty function method which allows convergence to be achieved without having to drive the weights to infinity (see Reference 2). Here, the performance index is defined as

$$J = P(X) + (1/2) \sum_{i=1}^m \sigma_i r_i^2(X, \theta_i) \quad (60)$$

where

$$r_i(X, \theta_i) = \begin{cases} C_i(X) - \theta_i & , \quad i = 1 \rightarrow K \\ \min [C_i(X) - \theta_i, 0] & , \quad i \in K + 1 \rightarrow M \end{cases} \quad (61)$$

and where σ_i is a positive, constant weight.

A description of the computational algorithm is as follows:

1. Guess initial values for X , θ_i , and σ_i . Since X represents the final time, the controls, and perhaps the ignition time, it should be easy to run some constant control trajectories and find a value for X which gives a low value to the performance index P . For the first run, the constants θ_i are usually set equal to zero. Finally, the weights σ_i can be obtained by requiring each constraint term in (60) to have the same order of magnitude as P .

2. Minimize the performance index with respect to X holding θ_1 and σ_1 constant. This step represents an unconstrained minimization which is performed with the Davidon-Fletcher-Powell method.
3. Assuming that $X = X(\theta)$, minimize the performance index with respect to θ , starting from the X which results from step (2). The purpose of this step is to find new values of θ such that the constraints are satisfied. Only a single iteration of the optimization is performed. For equality constraints, this is done with the Newton-Raphson method, while for inequality constraints, a minimization problem is formed which guarantees that the corresponding Lagrange multipliers remain nonnegative.
4. If the rate at which a constraint is being reduced is not sufficiently large, the value of the corresponding weight θ_1 is increased.
5. If convergence has not been achieved, go to 2.

4.2 Optimization Code

The code which has been selected for solving the optimization problem has been obtained from the AERE in Harwell, England. It is a general purpose code for solving the nonlinear programming problem using the augmented-Lagrangian method. The code contains five subroutines which are described briefly below.

VF01A: This subroutine directs the optimization process.

VA09A: This subroutine performs unconstrained minimization using the Davidon-Fletcher-Powell method. The metric is represented in a factored form to preserve positive definiteness so that a crude one-dimensional search can be used. Convergence is superlinear.

MC11A: This set of subroutines (MC11A, B, C, D, and E) performs matrix manipulations such as factorization and the metric update.

VE04A: This subroutine performs the optimization for satisfying the constraints.

VF01Z: This subroutine forms the augmented Lagrangian (60) and its derivative from the performance index (57), the equality constraints (58), the inequality constraints (59), and their derivatives.

The documentation which has been supplied with these subroutines is contained in Appendix A. Also, a listing of the code is presented in Appendix B. That part of the code which must be supplied by the user is shown in Appendix C.

The user must provide a subroutine entitled VF01B which computes the performance index F and the constraints C as well as the derivatives F_x and C_x . In order to minimize the amount of set-up time, it has been decided to compute the derivatives numerically, using central differences. This approach also allows the user to change the physical model without having to change derivative subroutines as well.

This optimization code has been verified by applying it to the solution of a number of problems whose solutions are known. These problems include a simple quadratic function in two variables with a linear equality or inequality constraint and the Rosenbrock function with a quadratic constraint. Also, the optimal control problem known as the lunar launch problem has been solved. The problem is to transfer a rocket from the surface of the moon to orbital conditions in minimum time, assuming that the ratio of the thrust to the mass is constant. The control variable is the angle between the thrust vector and the horizontal. To check out inequality constraints, an active upper limit has been imposed on the steering angle. In all cases, the Harwell code performed extremely well, and the known extremal solutions were obtained in a correct or reasonable number of iterations.

After solving the above problems, it became apparent that some minor modifications had to be made in the Harwell code. First, if an inequality constraint was imposed and was exactly equal to zero on the first iteration, the program aborted. To prevent this, the constraint was set equal to a small positive value. Second, for solving more complex problems, it became apparent that it would be necessary to interrupt the computation, store some results on a tape, and restart the computation. Such modifications have been made. Finally, for solving the reentry and plane change problems, some procedure for preventing the flight path inclination from becoming -90 deg had to be incorporated. If this happens, the heading angle equation blows up, and the program terminates. This situation occurs during the one-dimensional search associated with the unconstrained minimization. If too large a change is made in the parameters, the vehicle can end up in a vertical dive. The problem has been eliminated by monitoring the flight path inclination on every integration step. If it gets lower than -60 deg, the one-dimensional search is terminated, the search stepsize is reduced, and the search is restarted.

SECTION V

USER CODE

The user of the optimization code must provide a main program and a subroutine or sequence of subroutines in which the performance index, the constraints, and their derivatives are calculated. These parts of the computer program are discussed here. Because of the similarity of the two trajectory optimization problems, it has been possible to combine them into a single program. The listing of the user code is presented in Appendix C. The variables in the user code are defined in Appendix D.

5.1 Main Program

The main program MRRV performs two functions. First, all of the parameters needed by VF01A are defined (see Appendix A). Second, the process for interrupting the computation and storing intermediate numbers on tape as well as reading intermediate numbers from tape and restarting the computation is controlled.

5.2 Performance Index, Constraints, and Derivatives

Subroutine VF01B calls subroutine SG, which computes the performance index and the constraints, and subroutine SGX, which computes the derivatives of the performance index and the constraints.

In subroutine SG, the tables defining the control variable histories are formed from the current values of the parameters. The initial conditions for the differential equations are defined, and the differential equations of motion are integrated to the final time. From the final values of the states, the performance index and the constraint residuals are computed. While the trajectory is computed in nondimensional variables, the printout is in terms

of dimensional variables. For the plane change problem, the engine is ignited at t_b and shut off at $t_b + \Delta t_b$. Finally, if the flight path inclination becomes less than -60 deg at any point of the trajectory, the integration is stopped, and the computation is redirected to the one-dimensional search for a decrease in the search stepsize.

The integration is performed using a fixed-step, fourth-order, Runge-Kutta integration method (subroutine RUNGE). The number of integration steps has been chosen by making calculations of the penalized performance index and its derivatives for different step sizes and choosing the largest stepsize which gives reasonable accuracy. This has been done to minimize the computation time and to keep the integration as far away from round-off error as possible.

The integration subroutine makes repeated calls to the subroutine DERIV which contains the differential equations of motion of the vehicle and the differential equations for the inequality constraints. Here, the tables containing the controls are read; the aerodynamic, propulsion, and mass characteristics are determined; and the right-hand sides of the differential equations are computed. The aerodynamic characteristics are obtained from a call to the subroutine AERO which reads the aerodynamics tables. To obtain the atmospheric properties, AERO calls subroutine ATM62.

Subroutine SLIN1 performs linear interpolation of a table of data points. It has been provided to read the control tables and the aerodynamics tables.

The last subroutine supplied by the user is SGX. Here, the derivatives of the performance index and the constraints are computed by central differences. Hence, two function evaluations are made for each derivative computed.

SECTION VI
SOLUTION OF THE OPTIMIZATION PROBLEMS

The general procedure which has been followed in the solution of the optimization problems is to find a nonoptimal, constant control trajectory which gives a reasonable value to the performance index and is somewhat near the desired final conditions. This is done by varying the parameters (final time, angle of attack, bank angle, and ignition time if necessary) systematically until a reasonable trajectory is obtained. Then, these parameters are used as the initial guess of the optimal trajectory.

6.1 Reentry Problem

For the reentry problem, a reasonable set of nominal parameters has been found to be $t_f = 3200$ sec, $\alpha = 20$ deg, and $\mu = 60$ deg. The results from integrating the equations of motion for these values of the parameters are presented in Table 5. Note that the nominal crossrange is $\phi_f = 26.4$ deg and that the final condition is given by

$$C_1 = h_f/100,000 - 1 = 0.136 \quad (62)$$

The main program contains all of the input quantities and is presented in Appendix C for the reentry problem (IPROB = 1). The iteration process starts from the nominal path (IRS = 0), and no more than TMAX = 1200 sec of computer time are to be used on the first run. To allow the program to reach the point where the time check is made, the external time limit has been set at 1500 sec. A total of eleven parameters ($N = 11$) is being used, of which one is the final time, five ($NA = 5$) are the ordinates of the angle of attack table, and five ($NM = 5$) are the ordinates of the bank angle table. There is one constraint ($M = 1$) which is an equality constraint ($K = 1$). Next, the first

Table 5. Nominal Path for Maximum Crossrange

x_1 = 3.96785	x_2 = .349066	x_3 = .349066	x_4 = .349066
x_5 = .349066	x_6 = .349066	x_7 = 1.04720	x_8 = 1.04720
x_9 = 1.04720	x_{10} = 1.04720	x_{11} = 1.04720	

J = .373751E-1	F = -4.60469	C_1 = .135707
θ_1 = 0.	σ_1 = 54.0657	v_1 = 0.

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU	AP	MP	N
0.000	0.0	0.0	0.000	364800.	24500.	-1.20	0.00	20.000	60.00	0.000	0.000	.000
.100	320.0	21.22	.033	245196.	24517.	-.36	.51	20.000	60.00	0.000	0.000	.106
.200	640.0	42.18	.602	282646.	24042.	.49	2.07	20.000	60.00	0.000	0.000	.013
.300	960.0	62.89	1.311	321012.	23967.	.05	1.77	20.000	60.00	0.000	0.000	.001
.400	1280.0	83.59	1.845	293947.	23986.	-.84	1.17	20.000	60.00	0.000	0.000	.007
.500	1600.0	104.30	2.216	228427.	23671.	-.19	1.96	20.000	60.00	0.000	0.000	.210
.600	1920.0	125.37	3.495	236835.	22805.	-.21	4.50	20.000	60.00	0.000	0.000	.135
.700	2240.0	143.51	5.751	204459.	21202.	-.07	10.60	20.000	60.00	0.000	0.000	.437
.800	2560.0	160.35	10.419	180396.	18368.	-.04	23.54	20.000	60.00	0.000	0.000	.817
.900	2880.0	171.98	18.502	152426.	13508.	-.51	54.13	20.000	60.00	0.000	0.200	1.282
1.000	3200.0	172.82	26.383	113571.	5967.	-1.59	143.20	20.000	60.00	0.000	0.000	1.821

parameter $X(1)$ is set equal to the nominal final time, $X(2)$ through $X(6)$ are set equal to the nominal angle of attack history, and $X(7)$ through $X(11)$ contain the nominal bank angle history. Also, the values of the normalized time at each of the nodes are the same for each table, $TTA(I)$ and $TTM(I)$, and are given by 0.0, 0.25, 0.5, 0.75, and 1.0. The unconstrained minimization will terminate when the relative change in each parameter between two iterations is less than $EPS(I) = 1.E-4 * X(I)$. On the other hand, the constraint satisfaction iteration will terminate when the constraint residual satisfies the relation $C(1) \leq AKMIN * C(M + 1)$ where $AKMIN = 1.E-4$ and where $C(M + 1)$ is defined below. Since $C(M + 1) = 0.136$, an accuracy of 1.E-5 is being requested in the constraints.

The expected change in the performance index on the first iteration is $DFN = 0.5$. $MAXFN$ is set equal to a large number (10,000) because the program is now interrupted on a time basis rather than a function evaluation basis. Every iteration in VF01A is to be printed ($IPR1 = 1$) as is every iteration in VA09A ($IPR2 = 1$). The work space $W(I)$ is made to have $IW = 2500$ words as suggested by Appendix A. The optimization code will calculate the weight σ_1 ($MODE = 1$) using the constraint scale factor $C(M + 1) = 0.136$. To modify the code for interruption, it is now necessary to input the constraint scale factors in $C(M + 1)$ rather than in $C(I)$ as stated in Appendix A. Also, the parameter $AK = 1E60$ must be set here as well. The remaining quantities and the rest of the main program direct the writing on tape and reading from tape if the computation is interrupted (CPU time $> TMAX$).

A summary of the results is presented in Tables 6 though 9, and the characteristics of the converged trajectory are shown in Figures 1 through 3. Four cycles have been needed to achieve the desired accuracy in the final condition. It is interesting to note that a good trajectory is obtained after one cycle and that the remaining cycles are used to satisfy the final condition better. The

Table 6. Maximum Crossrange Trajectory: Cycle 1 Results

Iterations = 77		Function/Derivation Evaluations = 103	
X_1	= 3.16083	X_2	= .254757
X_5	= .260173	X_6	= .294001
X_9	= .781272	X_{10}	= .343563
X_4	= .258963	X_3	= .256750
X_8	= 1.07148	X_7	= .262842
X_{11}	= .429348E-1		
J	= -.8222392	F	= -.8222469
θ_1	= .161763E-2	σ_1	= .540657E+2
		C_1	= -.168584E-2
		v_1	= .874586E-1
TAU	TIME (SEC)	PHI (DEG)	ALTITUDE VELOCITY (FT) (FT/SEC)
0.000	0.0	0.000	364800. 24500.
.100	254.9	16.90	.005 256362.
.200	509.8	33.65	.620 163047.
.300	764.7	47.85	5.166 223073.
.400	1019.7	60.70	10.877 183957.
.500	1274.6	71.56	17.742 163588.
.600	1529.5	80.17	25.210 157960.
.700	1784.4	86.53	32.540 150807.
.800	2039.3	90.75	39.015 133654.
.900	2294.2	92.98	44.015 122599.
1.000	2549.2	93.74	47.124 99831.
		PSI (DEG)	ALPHA (DEG)
		0.00 -1.20	14.597 14.642
		.81 -.81	.11 .11
		.59 -.59	14.688 14.688
		.31 -.31	79.23 79.23
		20.90 14.736	58.07 58.07
		.99 30.23	0.000 0.000
		.91 39.00	51.41 44.837
		.75 48.60	44.76 44.76
		.22 .75	34.73 34.73
		57.77 14.893	0.000 0.000
		.75 66.85	16.24 15.295
		.11 76.56	0.000 16.070
		-2.88 84.10	0.000 16.846
			2.46

Table 7. Maximum Crossrange Trajectory: Cycle 2 Results

Iterations = 19

Function/Derivative Evaluations = 21

$x_1 = 3.16169$	$x_2 = .252030$	$x_3 = .257113$	$x_4 = .258866$
$x_5 = .260180$	$x_6 = .294529$	$x_7 = .262882$	$x_8 = 1.07297$
$x_9 = .780148$	$x_{10} = .345281$	$x_{11} = .419785E-1$	

$J = -.822246$

$F = -.822323$

$$\begin{aligned} \theta_1 &= .171019E-2 & c_1 &= -.810312E-4 \\ & & v_1 &= .924625E-1 \end{aligned}$$

$\theta_1 = .171019E-2$

$F = -.822323$

$$\begin{aligned} \theta_1 &= .171019E-2 & c_1 &= -.810312E-4 \\ & & v_1 &= .924625E-1 \end{aligned}$$

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU (DEG)	AP	MP	N
0.000	0.0	0.0	0.00	364800.	24500.	-1.20	0.00	14.440	150.62	0.000	0.000	.000
.100	255.0	16.90	.005	256357.	24606.	-.81	.11	14.557	14.96	0.000	0.000	.025
.200	510.0	33.66	.614	163162.	23247.	-.59	.37	14.673	79.31	0.000	0.000	.025
.300	765.0	47.87	5.153	223221.	20924.	-.30	.88	14.752	58.12	0.000	0.000	1.464
.400	1019.9	60.74	10.860	183834.	18966.	1.00	30.20	14.792	51.41	0.000	0.000	.120
.500	1274.9	71.61	17.722	163491.	16908.	.91	38.93	14.832	44.70	0.000	0.000	.453
.600	1529.9	80.24	25.185	157990.	14516.	.77	48.50	14.862	34.73	0.000	0.000	.713
.700	1784.9	86.61	32.514	150871.	11975.	-.77	57.67	14.892	24.77	0.000	0.000	.714
.800	2039.9	90.85	38.993	133756.	9172.	-.10	66.77	15.301	16.31	0.000	0.000	.6-5
.900	2294.9	93.10	44.000	122677.	5075.	-.10	76.49	16.088	9.36	0.000	0.000	.935
1.000	2549.8	93.87	47.116	99992.	3022.	-2.88	83.95	16.875	2.41	0.000	0.000	.912

Table 8. Maximum Crossrange Trajectory: Cycle 3 Results

Iterations = 4

Function/Derivative Evaluations = 6

$x_1 = 3.16159$	$x_2 = .252127$	$x_3 = .257088$	$x_4 = .258861$
$x_5 = .260161$	$x_6 = .294705$	$x_7 = .262870$	$x_8 = 1.07300$
$x_9 = .780257$	$x_{10} = .345261$	$x_{11} = .420158E-1$	

$$\begin{aligned} J &= -.8222237 & F &= -.8222315 & c_1 &= .169328E-4 \\ \theta_1 &= .169268E-2 & \sigma_1 &= 54.0657 & v_1 &= .915158E-1 \end{aligned}$$

Table 9. Maximum Crossrange Trajectory: Cycle 4 Results

Iterations = 1

Function/Derivative Evaluations = 2

X_1 = 3.16159	X_2 = .252132	X_3 = .257094	X_4 = .258865
X_5 = .260168	X_6 = .294708	X_7 = 2.62867	X_8 = 1.07301
X_9 = .780242	X_{10} = .345237	X_{11} = .421200E-1	
J = -.822239	F = -.822316	C_1 = .180706E-5	

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU (DEG)	AP	HP	N
0.000	0.0	0.0	0.000	364800.	24500.	-1.20	0.00	14.446	150.61	0.000	0.000	.000
.100	255.0	16.90	.005	256359.	24606.	-.81	.11	14.560	114.96	0.000	0.000	.036
.200	510.0	33.66	.614	163165.	23247.	-.59	.37	14.674	79.31	0.000	0.000	1.463
.300	764.9	47.87	5.153	223211.	20925.	-.30	.88	14.751	58.12	0.000	0.000	.120
.400	1019.9	60.73	10.859	183824.	18966.	1.00	30.20	14.791	51.41	0.000	0.000	.454
.500	1274.9	71.60	17.721	163482.	16908.	.91	38.93	14.832	44.70	0.000	0.000	.778
.600	1529.9	80.23	25.185	157883.	14516.	.75	48.51	14.862	34.74	0.000	0.000	.714
.700	1784.8	86.61	32.518	150868.	11974.	-.21	57.67	14.892	24.77	0.000	0.000	.646
.800	2039.8	90.85	38.993	133763.	9172.	-.75	66.77	15.302	16.31	0.000	0.000	.935
.900	2294.8	93.09	44.000	122684.	6075.	-.10	76.50	16.094	9.36	0.000	0.000	.912
1.000	2549.8	93.86	47.115	100000.	3051.	-2.88	83.97	16.886	2.41	0.000	0.000	.969

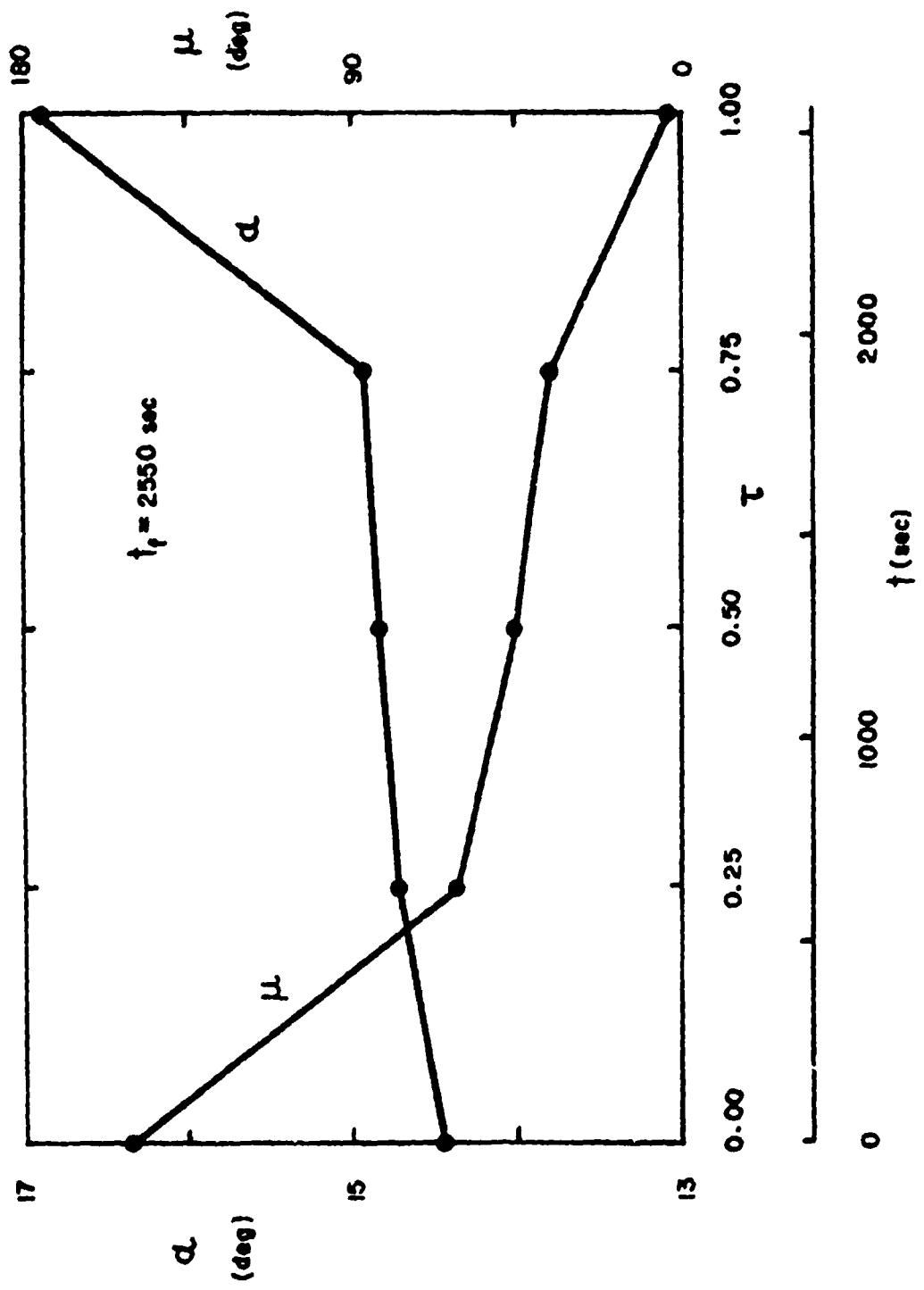


Figure 1. Maximum Crossrange: α and μ versus t

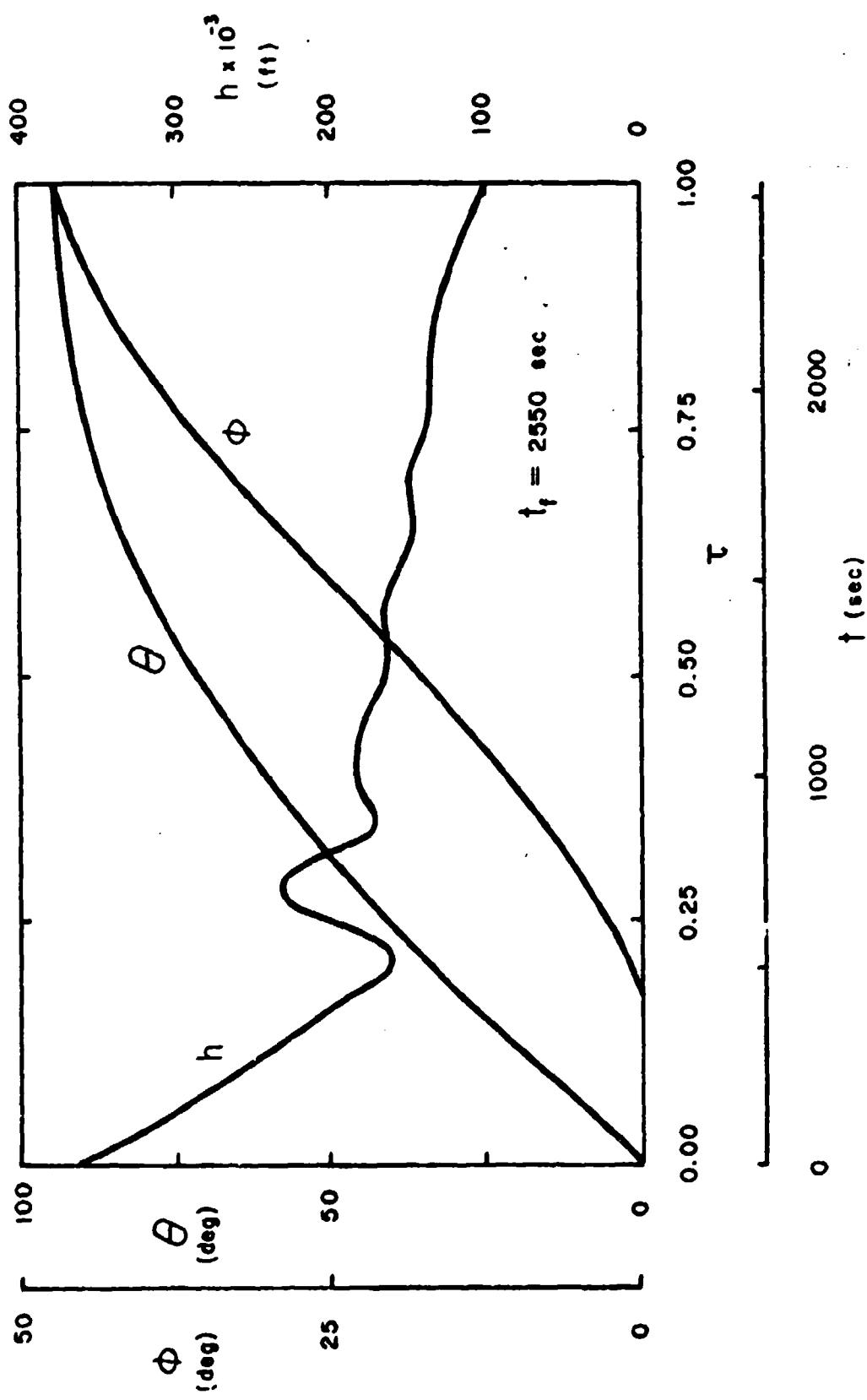


Figure 2. Maximum Crossrange: θ , ϕ , and h versus t

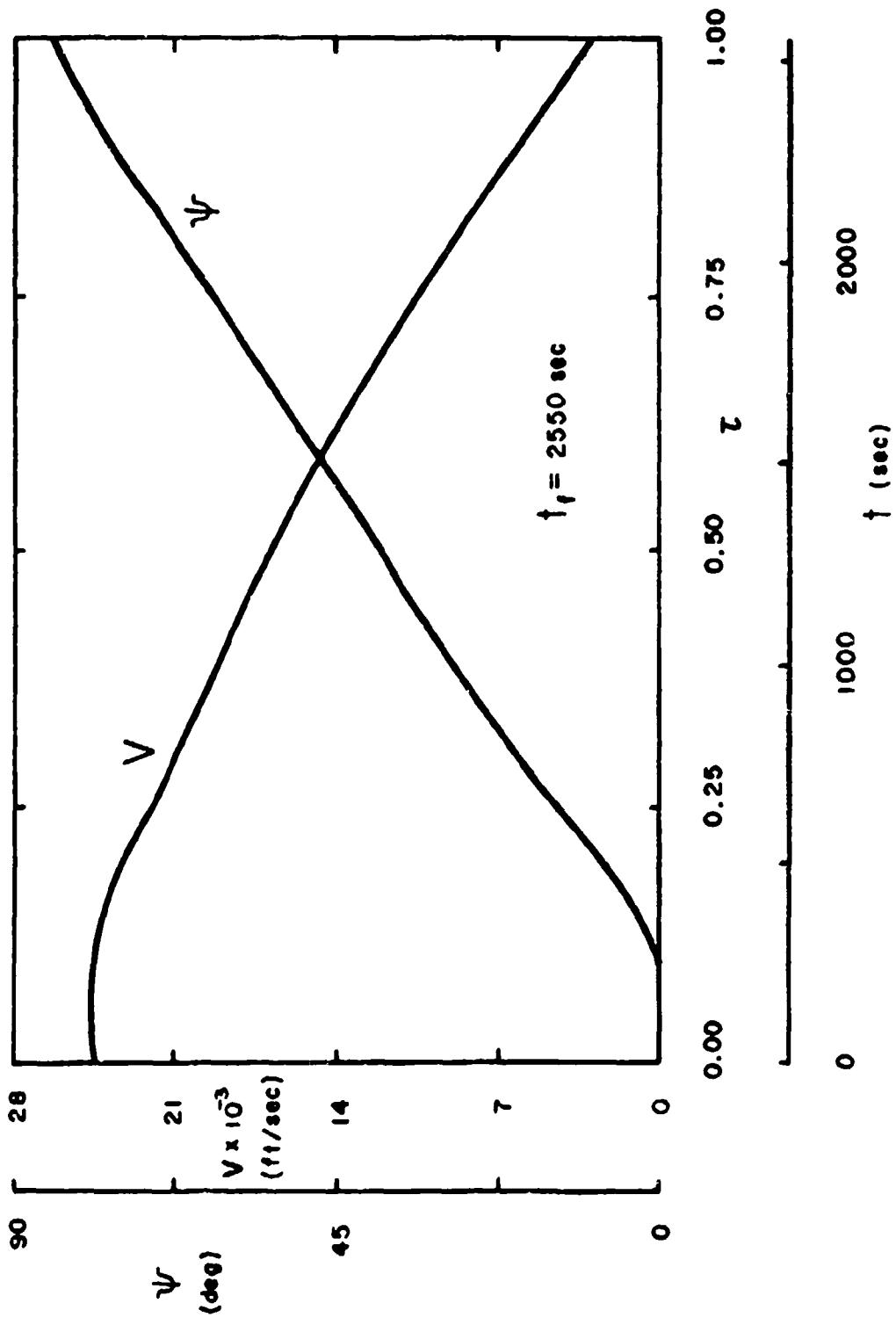


Figure 3. Maximum Crossrange: V and ψ versus t

optimal angle of attack seems to be at the value for maximum lift-to-drag ratio, although this has not been verified. With regard to the roll angle, the vehicle starts out at a high roll angle, so that it dives into the atmosphere, and then gradually reduces the roll angle until it is heading northward with a very small roll angle. The vehicle achieves a crossrange of approximately $\phi_f = 47$ deg and experiences a maximum load factor of around 1.5 g's.

It should be noted that the weight σ_1 has not been increased during the optimization process. This means that, after each cycle, sufficient progress has been made in converging to the constraints. On the other hand, θ_1 is changed during each cycle, and the Lagrange multiplier $v_1 \equiv \sigma_1 \theta_1$ appears to be approaching a limiting value. This multiplier and the optimal controls can be used to generate initial values of the Lagrange multipliers needed for starting the shooting method.

The complete computation required 1640 sec of CPU time (Cyber 170/750) and a total of 3000 function evaluations. This amounts to 0.55 sec per function evaluation, or 12.6 sec per function/derivative evaluation (23 function evaluations). The integration has been performed with 250 integration steps, and it may be possible to reduce the time by using forward differences. However, the convergence characteristics would deteriorate.

6.2 Plane Change Problem

For this problem, a good initial guess of the final time, the angle of attack, the bank angle, and the ignition time have been found to be $t_f = 2733$ sec, $\alpha = 40$ deg, $\mu = 60$ deg, and $t_b = 1624$ deg. The corresponding trajectory is shown in Table 10, from which it is seen that the orbital inclination is $i_f = 22.2$ deg and the constraint residuals are

Table 10. Nominal Path for Maximum Plane Change

$x_1 = 3.38900$	$x_2 = .698132$	$x_3 = .698132$	$x_4 = .698132$	$x_5 = .698132$
$x_6 = .698132$	$x_7 = 1.04720$	$x_8 = 1.04720$	$x_9 = 1.04720$	$x_{10} = 1.04720$
$x_{11} = 1.04720$	$x_{12} = 2.01400$		$j = .997474$	$F = .925560$
$c_1 = .863003E-2$	$c_2 = .882155E-2$	$c_3 = .109798E-1$	$c_4 = 0.$	$c_5 = 0.$
$\theta_1 = 0.$	$\theta_2 = 0.$	$\theta_3 = 0.$	$\theta_4 = 0.$	$\theta_5 = 0.$
$\sigma_1 = 617.284$	$\sigma_2 = 617.284$	$\sigma_3 = 413.223$	$\sigma_4 = 50000.0$	$\sigma_5 = 50000.0$
$v_1 = 0.$	$v_2 = 0.$	$v_3 = 0.$	$v_4 = 0.$	$v_5 = 0.$

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE VELOCITY (FT) (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU (DEG)	AP (DEG)	MP (DEG)	N (DEG)	I (DEG)
0.000	0.0	0.000	0.000	364800.	29500.	-1.20	0.00	60.00	0.000	0.000	0.000	0.000
.100	273.3	18.12	.002	252586.	24622.	-.68	-.05	60.00	-.000	0.000	.013	.042
.200	546.6	36.27	.114	217511.	24319.	.09	.82	60.00	-.000	0.000	.060	.781
.300	820.0	54.13	.485	243187.	23988.	.20	1.41	60.00	-.000	0.000	.019	1.807
.400	1093.3	71.81	.930	242717.	23860.	-.24	1.75	60.00	-.000	0.000	.020	1.651
.500	1366.6	89.39	1.428	192754.	23473.	-.55	2.12	60.00	-.000	0.000	.184	2.849
IGNITION OCCURRED AT 1625.25602 SECONDS												
.600	1639.9	106.04	2.675	160447.	21295.	-.05	7.66	60.00	-.000	0.000	.408	7.626
.700	1913.2	121.42	6.130	202020.	21675.	-.46	16.63	60.00	-.000	0.000	.123	16.681
.800	2186.5	137.66	11.719	322616.	24082.	1.83	20.19	60.00	-.000	0.000	.000	22.224
.900	2459.9	154.94	17.123	503034.	23869.	1.30	15.30	60.00	-.000	0.000	.000	22.237
1.000	2733.2	172.85	20.757	612427.	23707.	.63	8.69	60.00	-.000	0.000	.000	22.247

$$\begin{aligned}
 C_1 &\equiv h_f / 608,000 - 1 = .863003E-2 \\
 C_2 &\equiv v_f / 23,500 - 1 = .882155E-2 \\
 C_3 &\equiv \gamma_f = .109798E-1 \\
 C_4 &\equiv y_7 = .0 \\
 C_5 &\equiv y_8 = .0
 \end{aligned} \tag{63}$$

The last two constraints are the inequality constraint on the angle of attack ($\alpha \leq 40$ deg) and an inequality constraint on the bank angle ($\mu \geq 0$). In the solution of the minimization problem, the bank angle at the final point tried to go to a large negative value. The last inequality constraint has been included to keep this from happening.

The set-up for the plane change problem (IPROB = 2) is similar to that of the reentry problem. An additional parameter, the ignition time, is present making a total of twelve ($N = 12$). Here, there are five constraints ($M = 5$), of which three are equality constraints ($K = 3$). At first, the values of EPS had been chosen to be $1.E-4 * X(I)$. However, because of the extreme sensitivity of the problem to the ignition time, it has been necessary to use $EPS(I) = 1.E-5 * X(I)$. Finally, the constraint scale factors $C(M + I)$ for the altitude, the velocity, the flight path inclination, and the inequality constraints have been set equal to .009, .009, .011, .001, and .001, respectively. The first three values are rounded constraint residuals, and the last two have been chosen to produce relatively small weights.

A summary of the iteration process is presented in Tables 11 through 14, and the converged trajectory is illustrated in Figures 4 through 6. After four cycles, the constraints are satisfied to four significant figures. The converged angle of attack history stays near 40 deg (maximum C_L) at the

Table 11. Maximum Plane Change: Cycle 1 Results

Iterations = 39

Function/Derivative Evaluations = 72

$x_1 = 3.52859$	$x_2 = .633265$	$x_3 = .697462$	$x_4 = .272451$	$x_5 = .640335$
$x_6 = .573670$	$x_7 = 1.12206$	$x_8 = 1.19700$	$x_9 = 1.22940$	$x_{10} = .822619$
$x_{11} = .235914$	$x_{12} = 2.18744$	$j = .850941$	$F = .848507$	
$c_1 = -.260405E-2$	$c_2 = -.219832E-3$	$c_3 = -.125576E-2$	$c_4 = 0.$	$c_5 = 0.$
$\theta_1 = .268426E-2$	$\theta_2 = .573893E-4$	$\theta_3 = -.345124E-3$	$\theta_4 = 0.$	$\theta_5 = 0.$
$\sigma_1 = 617.284$	$\sigma_2 = 55812.4$	$\sigma_3 = 16640.4$	$\sigma_4 = 50000.0$	$\sigma_5 = 50000.0$
$v_1 = 1.65695$	$v_2 = 3.20303$	$v_3 = -5.74298$	$v_4 = 0.$	$v_5 = 0.$

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU	AP	HP	W	I (DEG)
0.000	0.0	0.0	0.000	364800.	24500.	-1.20	0.00	36.283	64.29	0.000	0.000	0.000	0.000
.100	284.6	18.87	.003	249204.	24625.	-.66	.05	37.754	66.01	0.000	0.000	.014	.051
.200	569.1	37.77	.144	213553.	24289.	.06	1.00	39.225	67.72	0.000	0.000	.068	.054
.300	853.7	56.31	.633	228726.	23885.	.02	1.86	35.090	68.95	0.000	0.000	.031	.085
.400	1138.3	74.65	1.284	198464.	23711.	-.56	2.27	25.350	69.70	0.000	0.000	.061	.085
.500	1422.9	92.75	2.407	124598.	22903.	-.35	6.49	15.610	70.44	0.000	0.000	.499	.537
.600	1707.4	109.30	5.886	143521.	20930.	-.25	15.82	24.042	61.12	0.000	0.000	.371	15.858
IGNITION OCCURRED AT 1764.37023 SECONDS													
.700	1992.0	124.17	11.648	183841.	20703.	.74	25.91	32.473	51.79	0.000	0.000	.156	26.700
.800	2276.6	139.89	19.381	23642.	2.37	27.34	35.925	40.41	0.000	0.000	.002	31.875	
.900	2561.2	157.96	26.644	523761.	23602.	1.33	19.36	34.397	26.96	0.000	0.000	.000	31.939
1.000	2845.7	177.73	31.006	606417.	23495.	.07	8.59	32.869	13.52	0.000	0.000	.000	31.950

Table 12. Maximum Plane Change: Cycle 2 Results

Iterations = 53

Function/Derivative Evaluations = 73

x_1 = 3.50225	x_2 = .697414	x_3 = .716539	x_4 = .235968	x_5 = .596629
x_6 = .707165	x_7 = 1.05933	x_8 = 1.20245	x_9 = 1.21915	x_{10} = .783545
x_{11} = -.379662E-1	x_{12} = 2.13215		J = .832534	F = .831345
c_1 = .211494E-2	c_2 = .453194E-4	c_3 = -.339378E-3	c_4 = -.207405E-3	c_5 = -.194383E-4
θ_1 = .714227E-3	θ_2 = .119134E-4	θ_3 = .813122E-4	θ_4 = .652144E-4	e_5 = 0.
σ_1 = 1894.59	σ_2 = .115433E+7	σ_3 = 270707.	σ_4 = 776031.	σ_5 = 200000.
v_1 = 1.35317	v_2 = 13.7521	v_3 = 2.20118	v_4 = 50.6084	v_5 = 0.

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU (DEG)	AP	MP	M	I (DEG)
0.000	0.0	0.00	0.000	364800.	24500.	-1.20	0.00	39.959	60.70	0.000	0.000	002	0.000
.100	282.4	18.73	.003	249891.	24621.	-.67	.06	40.397	63.98	-.008	0.000	015	.052
.200	564.9	37.47	.145	214478.	24252.	.05	1.01	40.836	67.26	-.078	0.000	069	.962
.300	847.3	55.83	.637	225028.	23823.	-.05	1.91	35.548	69.09	-.164	0.000	036	1.907
.400	1129.8	73.98	1.332	184216.	23584.	-.65	2.63	24.534	69.47	-.164	0.000	100	2.806
.500	1412.2	91.70	2.823	118632.	22430.	-.16	8.83	13.520	69.85	-.164	0.000	487	8.732
.600	1694.7	107.67	6.860	132674.	20409.	-.16	18.91	21.786	59.87	-.164	0.000	484	18.867
IGNITION OCCURRED AT 1719.54370 SECONDS													
.700	1977.1	122.08	13.323	181229.	20917.	.62	28.12	30.051	49.89	-.164	0.000	163	29.264
.800	2259.6	137.82	21.569	320314.	23877.	2.40	28.14	35.451	35.48	-.164	0.000	000	33.735
.900	2542.0	155.99	28.072	536670.	23595.	1.42	19.34	37.984	16.65	-.164	0.000	000	33.752
1.000	2824.5	176.08	33.060	609286.	23501.	-.02	7.68	40.518	-2.18	-.167	-.016	.000	33.763

Table 13. Maximum Plane Change: Cycle 3 Results

Iterations = 129

Function/Derivative Evaluations = 181

$x_1 = 3.59580$	$x_2 = .698234$	$x_3 = .699259$	$x_4 = .234367$	$x_5 = .529525$
$x_6 = .703480$	$x_7 = 1.17849$	$x_8 = 1.14165$	$x_9 = 1.25126$	$x_{10} = .680604$
$x_{11} = -.356270E-1$	$x_{12} = 2.22907$		$J = .828316$	$F = .826483$
$c_1 = .526879E-3$	$c_2 = .963806E-5$	$c_3 = .134132E-3$	$c_4 = -.141173E-5$	$c_5 = -.189220E-4$
$\theta_1 = .122092E-3$	$\theta_2 = .117335E-5$	$\theta_3 = -.457189E-4$	$\theta_4 = .530146E-6$	$\theta_5 = .800901E-5$
$\sigma_1 = 1894.59$	$\sigma_2 = .115433E+7$	$\sigma_3 = 117337.$	$\sigma_4 = .291760E+9$	$\sigma_5 = .115430E+7$
$v_0 = .231314$	$v_2 = 1.35443$	$v_3 = -5.36454$	$v_4 = 154.675$	$v_5 = 9.24482$

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU (DEG)	AP	MP	N	I (DEG)
0.000	0.0	290.0	19.23	364800.	24500.	-1.20	0.00	40.006	67.52	0.000	0.000	.000	0.000
.100		560.0	38.46	364800.	24500.	-.65	.06	40.029	66.68	-.000	0.000	.017	.061
.200		57.31	.686	247670.	24621.	-.08	1.09	40.053	65.83	-.000	0.000	.069	1.040
.300	870.0	75.95	1.378	227767.	23823.	-.03	1.91	34.737	66.67	-.001	0.000	.031	1.924
.400	1160.0	94.19	2.848	186754.	23622.	-.66	2.87	24.083	69.18	-.001	0.000	.038	2.701
.500	1450.0	110.49	7.149	116430.	22451.	-.20	8.87	13.428	71.69	-.001	0.000	.534	8.775
.600	1740.0	125.00	13.870	129300.	20331.	-.19	19.75	20.193	58.61	-.001	0.000	.494	19.694
IGNITION OCCURRED AT 1797.70812 SECONDS													
.700	2030.0	140.98	22.300	294961.	23909.	.51	28.75	26.957	45.53	-.001	0.000	.170	29.999
.800	2310.0	159.82	29.750	527771.	23605.	1.33	18.82	32.333	30.79	-.001	0.000	.002	34.233
.900	2610.0	180.71	33.780	608320.	23500.	.01	6.43	36.320	14.37	-.001	0.000	.000	34.250
1.0C.	2899.9							40.306	-2.04	-.001	-.015	.000	34.261

Iterations = 7

Function/Derivative Evaluations = 15

Table 14. Maximum Plane Change: Cycle 4 Results

X ₁	X ₂	X ₃	X ₄	X ₅
X ₆ = .700872	X ₇ = 1.18123	X ₈ = 1.14189	X ₉ = 1.24901	X ₁₀ = .684555
X ₁₁ = -.200911E-1	X ₁₂ = 2.22985		J = .827099	F = .826925
C ₁ = .512625E-4	C ₂ = .166243E-5	C ₃ = -.259077E-4	C ₄ = -.161095E-6	C ₅ = -.344284E-
θ ₁ = .104160E-4	θ ₂ = -.524920E-6	θ ₃ = -.166715E-4	θ ₄ = .706255E-7	θ ₅ = .131839E-
σ ₁ = 5776.30	σ ₂ = .516456E+7	σ ₃ = 117337.	σ ₄ = .664462E+10	σ ₅ = .288822E4
v ₁ = .601658E-1	v ₂ = -2.71098	v ₃ = -1.95619	v ₄ = 469.280	v ₅ = 38.0781

TAU	TIME (SEC)	THETA (DEG)	PHI (DEG)	ALTITUDE (FT)	VELOCITY (FT/SEC)	GAMMA (DEG)	PSI (DEG)	ALPHA (DEG)	MU	AP	MP	N	I (DEC)
0.000	0.0	0.00	0.000	364800.	24500.	-1.20	0.00	40.004	67.68	0.000	0.000	0.000	0.000
.100	290.2	19.25	.004	247611.	24621.	-.65	.07	40.010	66.78	-.000	0.000	.017	.061
.200	580.4	38.49	.166	213860.	24228.	.08	1.10	40.016	65.88	0.000	0.000	.069	1.043
.300	870.6	57.34	.689	227649.	23822.	-.03	1.92	34.694	66.65	-.000	0.000	.032	1.929
.400	1160.8	76.00	1.383	186482.	23620.	-.66	2.48	24.044	69.11	-.000	0.000	.089	2.711
.500	1451.0	94.00	2.859	116596.	22450.	-.19	8.87	13.394	71.56	-.000	0.000	.527	8.778
.600	1741.2	110.57	7.148	129476.	20338.	-.18	19.68	20.252	58.63	-.000	0.000	.492	19.634
IGNITION OCCURRED AT 1798.33299 SECONDS													
.700	2031.3	125.10	13.059	174174.	20596.	.51	28.68	27.111	45.69	-.000	0.000	.170	29.039
.800	2321.5	141.11	22.284	296410.	23906.	2.50	28.16	32.463	31.15	-.000	0.000	.002	34.188
.900	2611.7	159.97	29.724	528475.	23603.	1.32	18.77	36.310	15.00	-.000	0.000	.000	34.205
1.000	2901.9	180.87	33.741	608031.	23500.	-.00	6.39	40.157	-1.15	-.000	-.003	.000	34.216

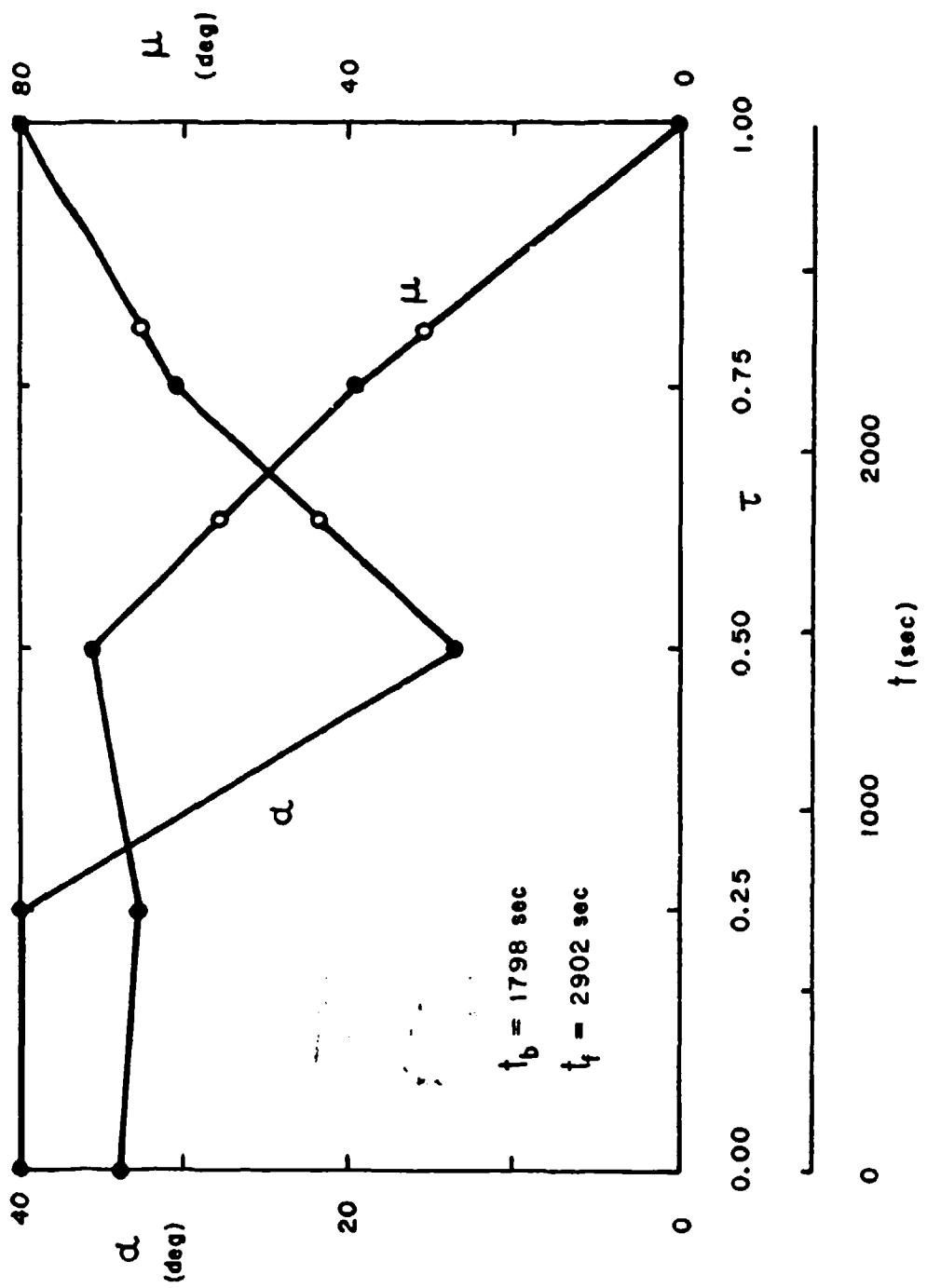


Figure 4. Maximum Plane Change: α and μ versus t

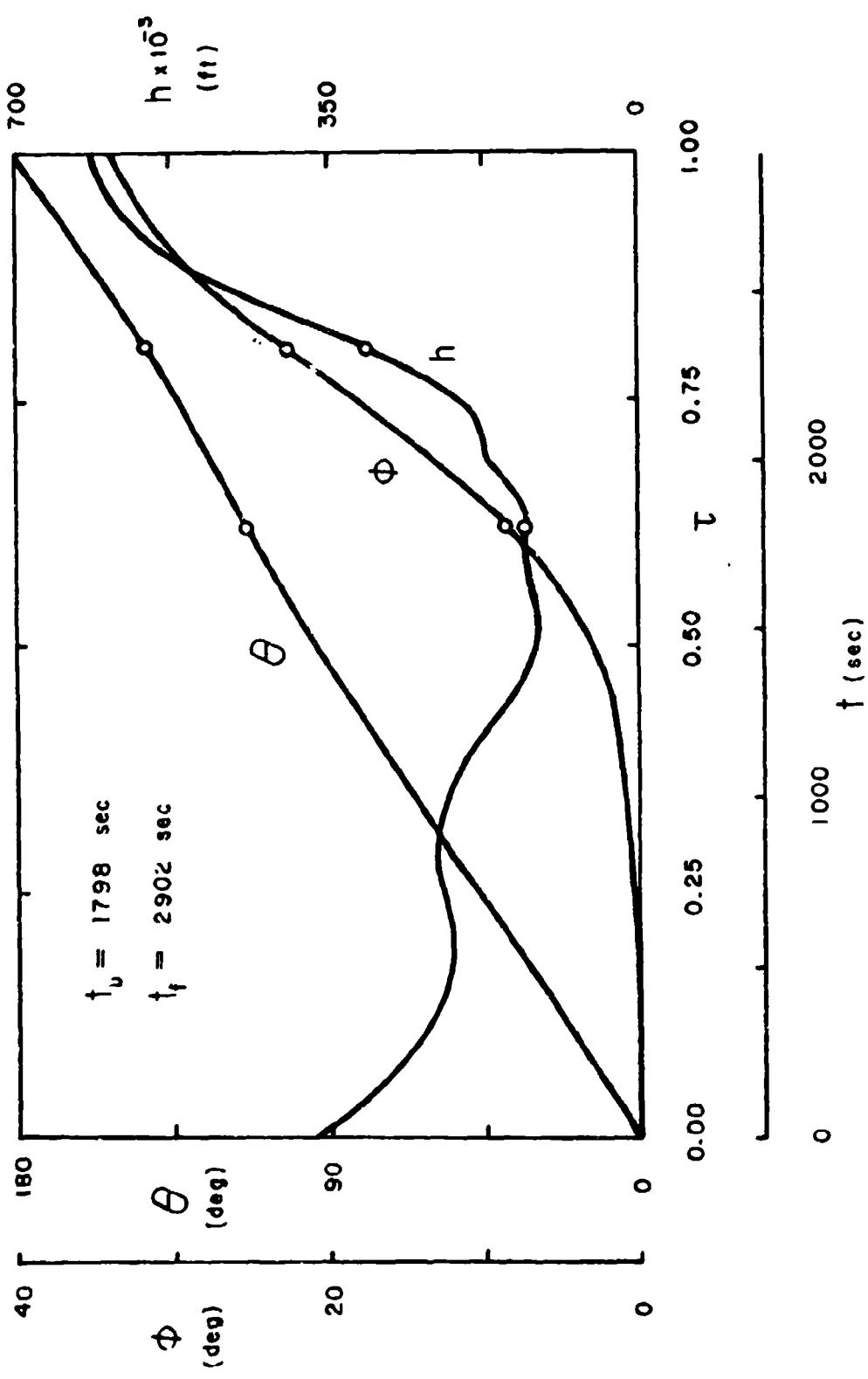


Figure 5. Maximum Plane Change: θ , ϕ , and h versus t

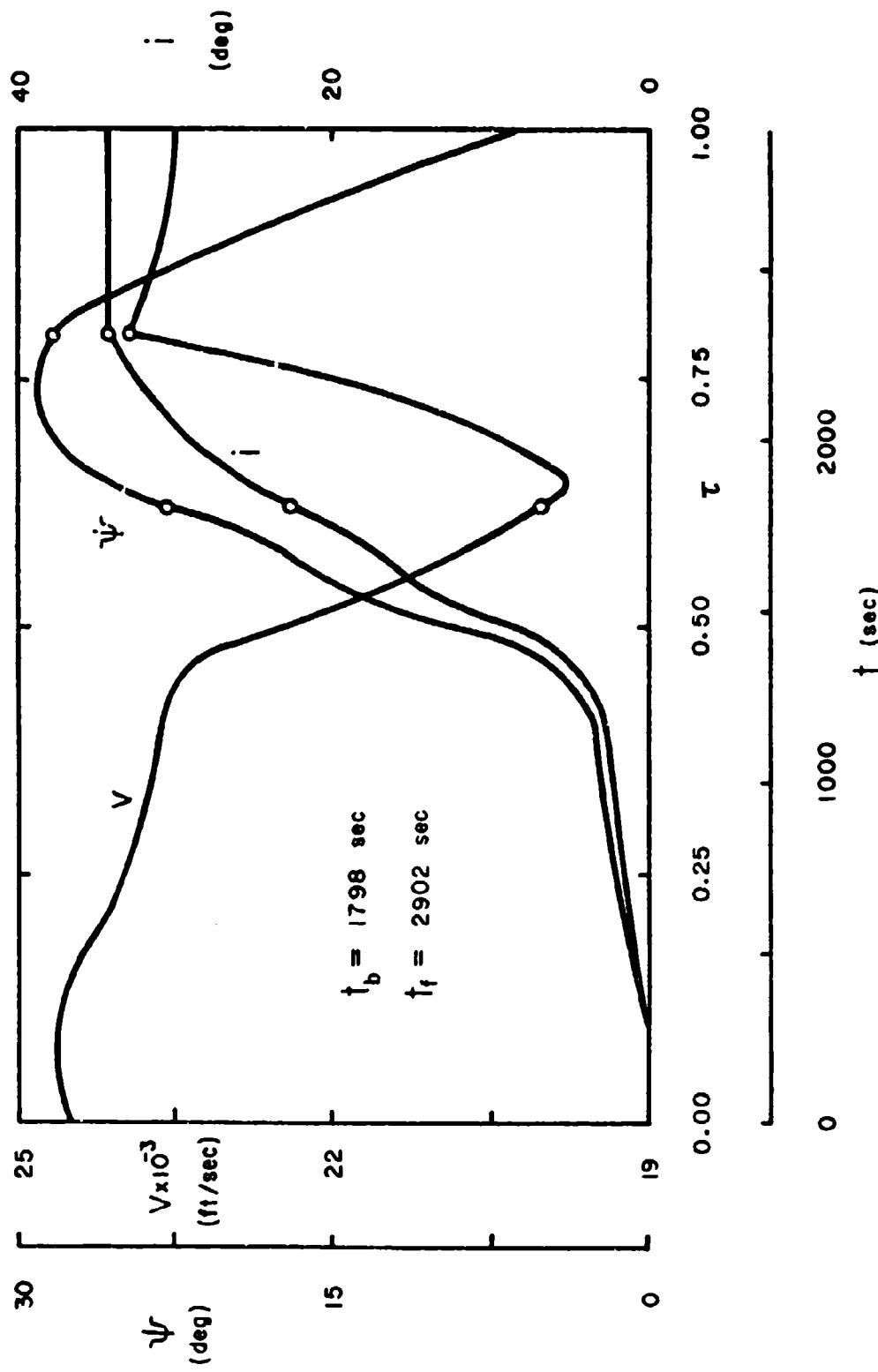


Figure 6. Maximum Plane Change: v , ψ , and i versus t

high altitudes and decreases to 13 deg (maximum C_L/C_D) at the low altitudes. On the other hand, the bank angle is near 70 deg during the entry part of the trajectory and decreases to 0 deg during the exit part. The maximum plane change is $i_f = 33.7$ deg, and the load factor does not exceed 0.5 g's.

The CPU time for the four cycles is 4175 sec on the CYBER 170/750 computer. This amounts to approximately 0.5 sec per function evaluation and 10 sec per derivative evaluation.

For this problem, the Lagrange multipliers ν do not seem to be approaching a limit. A possible reason is that the problem is so sensitive to the value of t_b that additional accuracy is needed in the numerical integration to achieve better convergence accuracy. Also, in generating numerical derivatives, the same perturbation size is used for each variable. More consistent results can be achieved by tailoring the perturbation size to each variable.

The behavior of the bank angle of the final point can be explained by assuming that the earth is not rotating. The initial point of the trajectory is located on the equator, and the vehicle is trying to put the velocity vector on the orbital plane with the highest inclination. As long as the longitude, θ , is less than 180 deg, the bank angle for maximizing the inclination is positive. However, once the vehicle crosses the equator into the southern hemisphere, it must bank in the opposite direction ($\mu < 0$) to obtain additional inclination. By restricting the bank angle to positive values, the maximum inclination should be obtained when $\theta_f = 180$ deg and $\mu = 0$. The optimal trajectory presented here has these features, as can be seen from Table 14. On the other hand, it should be possible to increase the inclination by relaxing the inequality constraint to $\mu \geq -90$ deg.

SECTION VII
DISCUSSION AND CONCLUSIONS

The maximum crossrange and the maximum plane change problems have been formulated as parameter optimization or nonlinear programming problems. An existing code for solving the nonlinear programming problem with the augmented-Lagrangian method has been selected and modified slightly for solving these problems. The derivatives needed for the optimization method have been computed numerically. As a consequence, the entire program can be viewed as a model for solving virtually any trajectory optimization problem.

A difficulty in solving trajectory optimization problems lies in the model used for the problem. For example, in performing the one-dimensional search, it is possible to change the parameters so much that resulting trajectories go to extremely high altitudes or extremely low altitudes. This has caused an exponential overflow in the atmosphere subroutine. In another case, the resulting trajectories ended up in a vertical dive, causing the numerical integration to blow up because of the singularity in the heading angle equation. Initially, this difficulty had been eliminated by removing the singularity, which means increasing the number of differential equations by one and increasing the numerical complexity of the program (time becomes a dependent variable). Later, by additional modification of the optimization code, it has become possible to monitor the flight path inclination and terminate the one-dimensional search when γ gets too small. Another difficulty arose by trying to modify the drag polar so that the inequality constraint on the angle of attack could be eliminated. The optimization process has found a way to use the change to its advantage and has produced an unrealistic trajectory. Finally, in the

plane change problem, the bank angle at the final point wants to go to a large negative value. At this point, it is not understood what aspect in the model makes it useful for this to happen. Hence, an inequality constraint has been imposed on the bank angle to keep it nonnegative.

Once the modeling problems have been eliminated, the reentry problem is relatively easy to solve. This is not the case for the plane change problem, as can be seen from the number of iterations and the computer time required. Possible reasons include the number of constraints involved, sensitivity of the problem to ignition time, and numerical accuracy, either in the integration or in the computation of derivatives. With regard to constraints, the plane change problem is easy to solve if the bank angle is the only control and the altitude is the only constraint at the final point. Adding the other final conditions reduces greatly the progress achieved on every iteration. Also, including the angle of attack as a control requires the use of the inequality constraint on α .

The particular method for computing the derivatives, central differences, has been chosen because it is easy to modify the model. (If the derivatives are computed from differential equations, any change in the model produces a change in the derivative equations.) Although it has not been done, it is possible to cut the computer time roughly in half by only integrating over that part of the trajectory which is affected by a perturbation. Also, the computer time can be reduced further by using forward differences, or one function evaluation per derivative calculation. This reduction in accuracy will affect the accuracy with which the constraint can be satisfied.

The manner in which the optimization problems have been solved is sufficient to demonstrate the ability of the optimization code to handle such problems. Questions concerning the number of nodes needed to obtain an accurate solution and the placement of these nodes are still open, but they can be answered with a moderate amount of computer time.

REFERENCES

1. Bryson, A. E., Jr., and Ho, Y-C, Applied Optimal Control, John Wiley and Sons, New York, 1975.
2. Fletcher, R., "An Ideal Penalty Function for Constrained Optimization", J. Inst. Maths. Appl., Vol. 15, pp. 319-342, 1975.

APPENDIX A

**DOCUMENTATION ASSOCIATED WITH
THE OPTIMIZATION CODE**

1. Purpose

To find the minimum of a function $F(\underline{x})$ of n variables \underline{x} subject to both equality constraints $c_i(\underline{x}) = 0$ $i = 1, 2, \dots, k$ and inequality constraints $c_i(\underline{x}) \geq 0$ $i = k + 1, \dots, m$ ($k = 0$ or $k = m$ is allowed but k must be $\leq n$). Derivatives of all functions with respect to \underline{x} must be provided, both the vector $(\partial F / \partial x_1, \partial F / \partial x_2, \dots, \partial F / \partial x_n)$ and the matrix whose i th column is $(\partial c_i / \partial x_1, \partial c_i / \partial x_2, \dots, \partial c_i / \partial x_n)$ for $i = 1, 2, \dots, m$. These functions and derivatives must be specified in a user subroutine VFO1B (see section 4). An initial estimate of the solution must be provided which need not be feasible. The subroutine allows advantage to be taken of the possibility that some of the constraints are linear, and also of certain other types of information about the problem, if available. If all the constraints are linear, the use of VFO1A is not most efficient, and one of the LA or VE routines should be used. The method is a penalty function - Lagrangian method (see section 8) and VFO1A calls VA09A to carry out the associated unconstrained minimizations.

2. Argument List

CALL VFO1A(N,M,K,X,EPS,AKMIN,DFN,MAXFN,IPR1,IPR2,IW,MODE)

N An INTEGER set to the number of variables n ($N \geq 2$).

M An INTEGER set to the total number of constraints m ($M \geq 1$).

K An INTEGER set to the total number of equality constraints k .

X A REAL array of N elements in which the initial estimate of the solution must be set. VFO1A returns the solution \underline{x} in X.

EPS A REAL array of N elements, in which the tolerances for the unconstrained minimizations must be set. EPS(I) should be set so that $EPS(I)/X(I) \approx AKMIN$, roughly speaking.

AKMIN A REAL number in which the relative error tolerance required in the constraint residuals must be set. VFO1A will exit when $\max\{|c_i(\underline{x})|\} / \text{scaling factor for } c_i \leq AKMIN$ for the active constraints $\{i\}$ (see section 7).

DFN A REAL number in which the likely reduction in $F(\underline{x})$ must be set. This is done in the same way as for VA09A, - see the VA09A specification sheet.

MAXFN An INTEGER in which the maximum number of calls of VFO1B on any one unconstrained minimization must be set. Roughly speaking 2 or 3 times MAXFN calls of VFO1B are likely to be made altogether.

IPR1 An INTEGER controlling the frequency of printing from VFO1A. Printing occurs every IPR1 iterations, except for details of increases to the c_i which are always printed. No printing at all occurs (except for error diagnostics) if IPR1 = 0. All printing controlled by IPR1 is suitably annotated.

VFO1A 1

IPR2 An INTEGER controlling the frequency of printing from VA09A. IPR2 should be set as described in the VA09A specification sheet.

IW An INTEGER giving the amount of storage available in COMMON/VFO1L/W(.). Set to 2500 unless wishing to change the restrictions (see Section 5).

MODE An INTEGER controlling the mode of operation of VFO1A. If any positive definite estimate is available of the hessian matrix of the penalty function, set |MODE| = 2 or 3, otherwise set |MODE| = 1 (see VA09A specification sheet). If estimates of the σ_i and θ_i parameters are available (see section 8) set MODE < 0, otherwise set MODE > 0. A normal setting for a one-off job with no information available is MODE = 1.

3. The named COMMON areas

Certain named COMMON areas must be declared and set on entry to VFO1A.

COMMON/VFO1E/C(150)	Set scale factors (>0) for the constraints in C(1), C(2), ..., C(M). Choose the magnitude of these scale factors to give an indication of the magnitude of the constraints evaluated about the initial approximation \underline{x} . If any constraints are violated by an amount greater in modulus than that which is set, then the setting is increased accordingly. These scale factors are transferred to C(M+1), C(M+2), ..., C(2M) by VFO1A.
COMMON/VFO1F/GC(25,50)	Set the derivatives of any linear constraints on entry rather than in VFO1B. This is the most efficient and the numbers are not disturbed. The manner of setting is described in section 4.
COMMON/VFO1G/T(150)	If MODE < 0 is used, then set the parameters $\theta_1, \theta_2, \dots, \theta_M$ in T(1), T(2), ..., T(M) and the parameters $\sigma_1, \sigma_2, \dots, \sigma_m$ in T(M+1), T(M+2), ..., T(2M). The meaning of these parameters may be found in the report TP552 - see section 8.
COMMON/VFO1I/G2P(325)	If MODE = 2 or 3 set the estimated hessian matrix of the penalty function in G2P(1), ..., G2P(N*(N+1)/2). The manner of setting is that described in the specification sheet of VA09A under the heading MODE.

4. The user subroutine VFO1B

The user must declare a subroutine headed

```
SUBROUTINE VFO1B(N,M,X)
REAL X(1)
COMMON/VFO1C/F
COMMON/VFO1D/G(50)
COMMON/VFO1E/C(150)
COMMON/VFO1F/GC(25,50)
```

This subroutine must take \underline{x} , the vector supplied in X(1), ..., X(N) and set $F(\underline{x})$ in F; $c_1(\underline{x}), \dots, c_m(\underline{x})$ in C(1), ..., C(M); $(\partial F / \partial x_1, \dots, \partial F / \partial x_n)_{\underline{x}}$ in G(1), ..., G(N); and set $(\partial c_1 / \partial x_1, \dots, \partial c_1 / \partial x_n)_{\underline{x}}$ in GC(1,I), ..., GC(N,I) for all $I = 1, 2, \dots, M$.

[Excepting the linear constraints which should be set on entry, as the numbers are constant]. Some time may also be saved if required by also including COMMON/VFO1G/T(150) and by not evaluating 'C(1,I),...,GC(N,I) when $C(I) \geq T(I)$ for any $I > K$. Note that the optimum values $F(x)$, $(\partial F/\partial x_1, \dots, \partial F/\partial x_n)$, etc. are left in these named COMMON areas on exit from VFO1A. Note also that in the double precision version the user subroutine name is VFO1BD and there is a D appended to the named COMMON areas.

5. Redefining named COMMON areas

Local storage for VFO1A is through named COMMON areas. These have been set on the assumption that $N \leq 25$ and $M \leq 50$. If it is desired to remove either or both of these restrictions, then it is necessary to increase the storage available in some or all of these areas. This can be done by defining the named COMMON areas in the users MAIN with the increased storage settings, in which case the extra storage will be effective throughout the whole program. The complete list of named COMMON used by VFO1A and the corresponding values of N and M are as follows.

COMMON/VFO1C/F,M,K,IS,MK,NU	independent of N and M
D/G(50)	2N
E/C(150)	3M
F/GC(25,50)*	N,M
G/T(150)	3M
H/GP(50)	μ ($\mu = \max(M,N)$)
I(G2P(325)	$N \cdot (N+1)/2$
J/V(50)	μ
K/WW(150)	3μ
L/W(2500)**	μ^2
M/ZZ(100)	2μ
N/LT(100)	$2M$

Notes:

- On increasing M , when $N < 25$, redefine GC with bounds (N,M) not $(25,M)$. VFO1A has been coded under this assumption, as it requires less storage. (VFO1A treats GC as a single suffix array).

**For M very large, μ^2 storage locations may be prohibitively large. However it is very unlikely that this amount of storage will actually be needed by VFO1A (no problem has yet been encountered for which more than $(2N)^2$ locations have been required). Hence in these circumstances, either declare W with $(2N)^2$ locations (or whatever can be spared), and set the integer IW to this number in the calling sequence of VFO1A. If more locations are required, then VFO1A will stop and print out the storage required.

6. General

Use of COMMON: named COMMON only - see section 5.

Workspace: 5K words unless N or M is redefined, when it is not usually more than $\sim 4\frac{1}{2}N^2 + NM + O(\max(M,N))$ words.

Other routines: Calls VFO1B (user subroutine), VFO1Z (private), VA09A, MC11A, MC11E. VA09A calls MC11B in addition.

Input/Output: No input, all output on stream 6 (line printer), controlled by user through IPR1 and IPR2.

Restrictions: $N \leq 25$ and $M \leq 50$ but can be lifted - see section 5.

Date of routine: August 1973

7. Accuracy

It may be that VF01A is unable to achieve the accuracy requested in the parameter AKMIN. In this case a diagnostic is printed. To find the cause of this, first examine the print out of the VA09A iteration. If this is anomalous ($\nabla \phi \rightarrow 0$ for instance) suspect a mistake in the programming of VF01B particularly in obtaining derivatives. If VA09A seems O.K., then other possible causes are (i) there is no feasible point (in which case $\sigma_i \rightarrow \infty$ and $c_i \rightarrow \text{const} \neq 0$), (ii) EPS has been set too large relative to AKMIN, (iii) AKMIN has been set to small relative to the machine precision, (iv) the problem is too ill-conditioned.

8. Method

That described in R. Fletcher, "An ideal penalty function for constrained optimization", C.S.S.2, December, 1973. The penalty function for inequalities is

$$\delta(\underline{x}, \underline{\theta}, \underline{\sigma}) = F(\underline{x}) + \frac{1}{2} \sum_i \sigma_i (c_i(\underline{x}) - \theta_i)_+^2$$

and each iteration involves minimizing $\delta(\underline{x})$ for fixed $\underline{\theta}, \underline{\sigma}$. After each iteration the $\underline{\theta}$ and $\underline{\sigma}$ parameters are varied so that the sequence of minima $\underline{x}, \underline{\theta}, \underline{\sigma}$ tends to the solution of constrained problem. The value of the product $\theta_i \sigma_i$ tends to the i th Lagrange multiplier of the problem. Any information about Lagrange multipliers, or about the hessian of δ can usefully be incorporated.

Convergence is guaranteed (in exact arithmetic) and this implementation of the method can be expected to converge at a second order rate.

October, 1973

4 VF01A

1. Purpose

To find the minimum of a function $F(x)$ of several variables, given that the gradient vector $(\partial F / \partial x_1, \partial F / \partial x_2, \dots, \partial F / \partial x_n)$ can be calculated.

The subroutine replaces VA01A to which it is superior in various ways (see section 5), and should be used whenever derivatives can be evaluated readily. It should however not be used either if storage space is at a premium (use VA08A) or if the function is a sum of squares (use VA07A). The subroutine complements VA08A, the latter requires four times the storage, and some comparisons (R. Fletcher, A.E.R.E. Report, R7125 (1972)) indicate that VA08A is marginally slower and more affected by round off error. As VA08A is more difficult to use, it is suggested that VA09A should be used in the first instance on any problem. If VA09A fails then VA08A should be tried as it is guaranteed to converge if the effect of rounding errors can be neglected.

2. Argument List

CALL VA09A(FUNCT,N,X,F,G,H,W,DFN,EPS,MODE,MAXFN,IPRINT,IEXIT)

FUNCT An IDENTIFIER of the users subroutine - see section 3.

N An INTEGER to be set to the number of variables ($N \geq 2$).

X A REAL ARRAY of N elements in which the current estimate of the solution is stored. An initial approximation must be set in X on entry to VA09A and the best estimate obtained will be returned on exit.

F A REAL number in which the best value of $F(x)$ corresponding to X above will be returned.

G A REAL ARRAY of N elements in which the gradient vector corresponding to X above will be returned. Not to be set on entry.

H A REAL ARRAY of $N^2(N+1)/2$ elements in which an estimate of the hessian matrix $\partial^2 F / \partial x_i \partial x_j$ is stored. The matrix is represented in the product form LDL^T where L is a lower triangular matrix with unit diagonals and D is a diagonal matrix. The lower triangle of L is stored by columns in H excepting that the unit diagonal elements are replaced by the corresponding elements of D. The setting of H on entry is controlled by the parameter MODE (q.v.).

W A REAL ARRAY of $3 \times N$ elements used as working space.

DFN	A REAL number which must be set so as to give VA09A an estimate of the likely reduction to be obtained in $F(\underline{x})$. DFN is used only on the first iteration so an order of magnitude estimate will suffice. The information can be provided in different ways depending upon the sign of DFN which should be set in one of the following ways:
	DFN>0 the setting of DFN itself will be taken as the likely reduction to be obtained in $F(\underline{x})$.
	DFN=0 it will be assumed that an estimate of the minimum value of $F(\underline{x})$ has been set in argument F, and the likely reduction in $F(\underline{x})$ will be computed according to the initial function value.
	DFN<0 a multiple $ DFN $ of the modulus of the initial function value will be taken as an estimate of the likely reduction.
EPS	A REAL ARRAY of N elements to be set on entry to the accuracy required in each element of X.
MODE	An INTEGER which controls the setting of the initial estimate of the hessian matrix in the parameter H. The following settings of MODE are permitted.
	MODE=1 An estimate corresponding to a unit matrix is set in H by VA09A.
	MODE=2 VA09A assumes that the hessian matrix itself has been set in H by columns of its lower triangle, and the conversion to LDL^T form is carried out by VA09A. The hessian matrix must be positive definite.
	MODE=3 VA09A assumes that the hessian matrix has been set in H in product form. This is convenient when using the H matrix from one problem as an initial estimate for another, in which case the contents of H are passed on unchanged.
MAXFN	An INTEGER set to the maximum number of calls of FUNCT permitted.
IPRINT	An INTEGER controlling printing. Printing occurs every $ IPRINT $ iterations and also on exit, in the form Iteration No, No of calls of FUNCT, IEXIT (on exit only) Function value X(1),X(2),...,X(N) 8 to a line (5 in VA09AD) G(1),G(2),...,G(N) 8 to a line (5 in VA09AD)
	The values of X and G can be suppressed on intermediate iterations by setting IPRINT<0. All intermediate printing can be suppressed by setting IPRINT=MAXFN+1. All printing can be suppressed by setting IPRINT=0.

IEXIT An INTEGER giving the reason for exit from VA09A. This will be set by VA09A as follows

IEXIT=0 (MODE=2 only). The estimate of the hessian matrix is not positive definite.

IEXIT=1 The normal exit in which $|DX(I)| < EPS(I)$ for all $I=1,2,\dots,N$, where $DX(I)$ is the change in X on an iteration.

IEXIT=2 $G^T DX > 0$. Not possible without rounding error. Probable cause is that EPS is set too small for computer word length.

IEXIT=3 FUNCT called MAXFN times.

3. User Subroutine

The user must provide a subroutine headed

SUBROUTINE XXX(N,X,F,G)

REAL X(1),G(1) (REAL*8 in VA09AD)

where XXX is an identifier chosen by the user.

This subroutine should use the variables x supplied in X(1), X(2), ..., X(N) to evaluate the function and gradient vector and place them in F and G(1), G(2), ..., G(N) respectively. XXX must be passed to VA09A as VA09A's first argument, see section 2, and appear in an EXTERNAL statement in the program that calls VA09A.

4. General

Use of COMMON : none

Workspace: $N^*(N+1)/2$ words + $4N$ words provided by the user in H and W.

Other routines: calls MC11A, B, E.

Input/Output: controlled by the user through IPRINT. All output is on stream 6 (line printer).

Restrictions: none

System dependence: none

Date of routine: April, 1972.

5. Method

The method used is a quasi-Newton method described by Fletcher (Computer Journal, Vol. 13, p.317, 1970), and is a modification of earlier methods of this type such as that implemented by VA09A. The method is superior to that in VA09A on three counts.

- (1) It uses a formula to update the hessian approximation H which has proved to be more efficient and reliable.
- (2) It uses a 'crude' line search which has been shown to be more efficient than an 'accurate' line search.
- (3) It represents H by the product LDL^T , which enables the positive definiteness of H to be guaranteed, even in the presence of round-off error.

1. Purpose

MC11A is a subroutine for use in problems which involve the addition or subtraction of rank one matrices $\sigma \underline{z}\underline{z}^T$ to positive definite, or semi-definite symmetric matrices A stored in factored form $A = LDL^T$, such that the resulting $N \times N$ matrix

$$\tilde{A} = 'A + \sigma \underline{z}\underline{z}^T$$

is also known to be positive definite or semi-definite. Note that L is lower triangular with $l_{11} = 1$, and D is diagonal with $d_1 \geq 0$. Apart from its obvious use in updating matrices which remain strictly positive definite, MC11A can be used (i) to accumulate a sum of rank one terms into an initial matrix $A=0$, (ii) to carry out projection and allied operations on A which reduce the rank, and (iii) to update matrices A of rank $k < n$ where it is known from other considerations that the rank remains unchanged. All these operations preserve the correct rank and are not seriously affected by round-off error. The method is that described by M.J.D. Powell and R. Fletcher (1973), 'On the updating of LDL^T factorizations', T.P. 519.

The matrix A is represented using the minimal storage of $N^*(N+1)/2$ elements where N is the dimension of the problem. To facilitate operating with A , a number of independent subroutines have been provided, written as entry points MC11B/C/.../F. These operations include reducing a matrix to its factors, multiplying out the factors, operating with the factors of A on a vector \underline{z} to obtain either $A\underline{z}$ or $A^{-1}\underline{z}$, and replacing the factors of A by the matrix A^{-1} . These facilities are described in more detail in section 4.

2. Argument List

CALL MC11A(A,N,Z,SIG,W,IR,MK,EPS)

- A A REAL one dimensional array of $N^*(N+1)/2$ elements in which the matrix $A=LDL^T$ must be given in factored form. The order in which elements of L and D are stored is $d_1, l_{21}, l_{31}, \dots, l_{N1}, d_2, l_{32}, \dots, l_{N2}, \dots \dots$
 $d_{N-1}, l_{N,N-1}, d_N$. The factors of the matrix
 $\tilde{A} = A + \sigma \underline{z}\underline{z}^T$ will overwrite those of A on exit.
- N An INTEGER ($N \geq 1$) which must be set to the dimension of the problem.
- Z A REAL one dimensional array of N elements in which the vector \underline{z} must be set. The array Z is overwritten by the routine.
- SIG A REAL variable in which the scalar σ must be set. SIG is not restricted to ± 1 , but if $SIG < 0$ then it must be known from other considerations that \tilde{A} is positive definite or semi-definite, apart from the effects of round-off error.

W A REAL array of N elements. If $SIG > 0$ then W is not used, and the name of any one dimensional array can be inserted in the calling sequence. If $SIG < 0$ then W is used as work space. In addition for $SIG < 0$ it may be possible to save time by setting in W the vector \mathbf{v} defined by $\mathbf{Ly} = \mathbf{z}$. The ways in which this can occur are described under MK below.

IR An INTEGER to be set so that $|IR|$ is the rank of \mathbf{A} . If the rank of $\tilde{\mathbf{A}}$ is expected to be different from that of \mathbf{A} , set $IR \neq 0$. On exit from MC11A, $IR(0)$ will contain the rank of $\tilde{\mathbf{A}}$.

MK An INTEGER to be set only when $SIG < 0$, as follows. If the vector \mathbf{v} defined by $\mathbf{Ly} = \mathbf{z}$ has not been calculated previously, set $MK = 0$. If MC11E has been used previously to calculate $\mathbf{A}^{-1} \mathbf{z}$, then \mathbf{v} is a by-product of this calculation and is stored in the W parameter of MC11E. In this case transfer \mathbf{v} to the W parameter of MC11A and set $MK = 1$. If \mathbf{z} has been calculated as $\mathbf{z} = \mathbf{Au}$ for some arbitrary vector \mathbf{u} using MC11D, then again \mathbf{v} is a by-product of the calculation and is available in the W parameter of MC11D. In this case (or any other in which \mathbf{v} is known) set \mathbf{v} in the W parameter of MC11A and set $MK = 2$.

EPS A REAL variable to be set only when $SIG < 0$ and $\tilde{\mathbf{A}}$ is expected to have the same rank as \mathbf{A} . In certain ill-conditioned cases a non-zero diagonal element of $\tilde{\mathbf{D}}$ ($\tilde{\mathbf{A}} = \mathbf{LDL}^T$) might become so small as to be indeterminate. Two courses of action are possible. One is to introduce a small perturbation in order that $\tilde{\mathbf{A}}$ keeps the same rank as \mathbf{A} . This is the normal course of action and is achieved by setting EPS equal to the relative machine precision ϵ . ϵ is $\sim 10^{-7}$ in single length arithmetic and $\sim 10^{-18}$ in double length on the IBM 370. The other course of action is to let the rank of $\tilde{\mathbf{A}}$ be one less than the rank of \mathbf{A} . This is achieved by setting EPS equal zero.

3. General

Use of COMMON: none

Workspace: $N^2(N+1)/2 + N + N$ words provided by the user in A, Z and W. If $SIG > 0$ W need not be supplied.

Entry points: MC11B/C/.../F - see section 4.

Other routines: none.

Input/Output: none.

Timing: One call of MC11A requires $\sim n^2$ multiplications, unless $SIG < 0$ and $MK = 0$ when the figure is $\sim 1\frac{1}{2} n^2$. One call of any of MC11B, C or F requires $\sim n^3/6$ multiplications. One call of either MC11D or E requires $\sim n^2/2$ multiplications.

Restrictions: none.

System dependence: none.

Date of routine: January, 1973.

4. Other Entry points

Other entry points are provided to facilitate operating with A which is stored in compact form. In all of these A is a REAL one dimensional array of $N(N+1)/2$ elements, and $N \geq 1$ is an integer set to the dimension of the problem. Each entry point is essentially an independent subroutine, and could be taken out and written as such if desired.

MC11B - factorize a positive definite symmetric matrix given in A.

CALL MC11B(A,N,IR)

- A Must contain the elements of A in the order $a_{11}, a_{21}, \dots, a_{N1}, a_{22}, a_{32}, \dots, a_{N2}, \dots, a_{N-1, N-1}, a_{N, N-1}, a_{NN}$; (that is as successive columns of its lower triangle). On exit A will be overwritten by the factors L and D in the form described in section 2 under A.
- N Order of the matrix A.
- IR An INTEGER set by MC11B to the rank of the factorization. If the factorization has been performed successfully IR=N will be set. If IR<N then the factorization has failed because A is not positive definite (possibly due to round-off error). In this case the factors of a positive semi-definite matrix of rank IR will be found in A. However the results of this calculation are unpredictable, and MC11B should not be used in an attempt to factorize positive semi-definite matrices.

MC11C - multiply out the factors LD^T to obtain A.

CALL MC11C(A,N)

- A Must be set in the factored form described in section 2 under A. On exit the factors will be overwritten by the explicit matrix A, the order of the elements being that described for input to MC11B.
- N Order of the matrix A.

MC11D - calculate the vector $\underline{z}^* = A\underline{z}$ where A is in factored form.

CALL MC11D(A,N,Z,W)

- A Must be set in factored form.
- N Order of the matrix A.
- Z A REAL array of N elements to be set to the vector \underline{z} . On exit \underline{z} contains the vector $\underline{z}^* = A\underline{z}$.
- W A REAL array of N elements which is set by MC11D to the vector \underline{v} defined by $L\underline{v} = \underline{z}^*$. If this vector is not of interest, replace \underline{W} by \underline{Z} in the calling sequence to obviate the need to supply extra storage.

MC11E - calculate the vector $\underline{z}^* = A^{-1} \underline{z}$ where A is in factored form.

CALL MC11E(A,N,Z,W,IR)

- A Must be set in factored form.
- N Order of the matrix A.
- Z A REAL array of N elements to be set to the vector \underline{z} . On exit Z contains the vector $\underline{z}^* = A^{-1} \underline{z}$.
- W A REAL array of N elements which is set by MC11E to be vector \underline{v} defined by $L \underline{v} = \underline{z}$. If this vector is not of interest, replace W by Z in the calling sequence to obviate the need to supply extra storage.
- IR An INTEGER which must be set to the rank of A.

MC11F - calculates the explicit matrix A^{-1} from the factors of A.

CALL MC11F(A,N,IR)

- A Must be set in factored form. MC11F will overwrite this by the elements of the inverse matrix A^{-1} , in the order $a_{11}^{-1}, a_{21}^{-1}, \dots, a_{NN}^{-1}$ as for MC11B.
- N Order of the matrix A.
- IR An INTEGER which must be set to the rank of A.

Notes. (i) MC11F should not be used to solve equations, in which case MC11E should be used. MC11F is intended for applications in which the explicit elements of A^{-1} must be examined, for example in the use of variance-covariance matrices. (ii) MC11E and F will RETURN without calculation unless IR=N.

January, 1973

1. Purpose

To find the value of $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ which minimizes a quadratic function $Q(\mathbf{x})$ of n variables \mathbf{x} , subject to upper and lower bounds on some or all of the variables. $Q(\mathbf{x})$ is defined by

$$Q(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{b}^T \mathbf{x} = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n x_i A_{ij} x_j - \sum_{i=1}^n b_i x_i \quad (1)$$

where \mathbf{A} is symmetric ($A_{ij} = A_{ji}$), and the bounds are of the form $l_i \leq x_i \leq u_i$. It is permissible to let $l_i = u_i$ if required, and \mathbf{A} is not restricted to being positive definite. The subroutine calculates the solution $\mathbf{x} = \xi$, the minimum value $Q(\xi)$, and the gradient $\mathbf{g}(\xi)$ (note $\mathbf{g}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$). This problem is a special case of quadratic programming for which a subroutine VEO2A exists. VEO4A is more efficient and reliable for solving problems of this special form.

An application of the subroutine is to (weighted) linear least squares data fitting subject to bounds. If it is required to minimize

$$S(\mathbf{x}) = (\mathbf{B}\mathbf{x} - \mathbf{y})^T \mathbf{W} (\mathbf{B}\mathbf{x} - \mathbf{y}) \quad (2)$$

subject to the above bounds, where \mathbf{B} is an $m \times n$ matrix $m \geq n$ and \mathbf{W} is an $m \times m$ diagonal matrix of weights $W_{ii} > 0$, then set $\mathbf{A} = 2\mathbf{B}^T \mathbf{W} \mathbf{B}$ and $\mathbf{b} = 2\mathbf{B}^T \mathbf{W} \mathbf{y}$. Statistical calculations for this problem are described in section 5, including an additional entry point to enable the variance-covariance matrix to be calculated.

2. The Argument list

CALL VEO4A (N,A,IA,B,BL,BU,X,Q,L,I,K,G)

N an INTEGER which must be set by the user to the number of variables.

A a REAL, two dimensional array, each dimension at least N; the elements in the upper triangle $A(I,J)$ $I \leq J \leq N$ must be set by the

VE04A 1

user to the corresponding A_{ij} in (1), and will remain untouched by the subroutine. Elements $A(I,J) \text{ } N > I > J$ are used as working space.

IA an INTEGER giving the first dimension of A in the statement which assigns space to A.

B a REAL array of at least N elements. The user must set B(I) to the b_i in (1). B is not overwritten by VEO4A.

BL a REAL array of at least N elements. The user must set BL(I) to the lower bound l_i on the I^{th} variable. If the bound is non-existent, set l_i to a very small number like -1E75. BL is not overwritten by VEO4A.

BU a REAL array of at least N elements. The user must set BU(I) to the upper bound u_i on the I^{th} variable. If the bound is non-existent, set u_i to a very large number. BU is not overwritten by VEO4A.

X a REAL array of at least N elements. VEO4A returns the solution ξ_i in X(I).

Q a REAL variable in which VEO4A returns the value of $Q(\xi)$.

LT an INTEGER array of at least N elements, set by VEO4A to a permutation of the integers 1,2,...,N (see K and G below)

K an INTEGER set by VEO4A to the number of free variables at the solution ξ (those not on their bounds). These are the variables LT(1), LT(2),...,LT(K).

G a REAL array of at least 3*N elements. G(1),...,G(N) are set by VEO4A to the gradient $g(\xi)$. G is indirectly addressed so that G(I) contains the gradient with respect to the LT(I) variable, whence G(1),...,G(K) will be found to be zero. G(N+1),...,G(3*N) are used by VEO4A as working space.

3. General Information

Use of COMMON: none

Workspace: approx $\frac{1}{2}n^2$ words (half of A) + 2n words in C and n integers in LT.

Other routines: none

Entry points: VEO4B-see section 5

Input/Output: none

System dependence: none

Timing. The time required depends upon how many free variables k there are at the solution. Typical figures for (k/n, number of multiplications) are (.1, $n^3/12$), (.5, $n^3/6$), (.75, $n^3/2$).

Restrictions: none

Date of Routine: April 1973.

4. Method

That of Fletcher R. and Jackson, M.P., (1973), "Minimization of a quadratic function of many variables subject only to lower and upper bounds", T.P.528.

This method combines generality (any A), efficiency (times comparable to those required for a factorization of A) and stability (uses partial LDL^T factorizations).

5. Statistical Calculations

When a sum of squares is being minimized as in (2), then certain statistical quantities can readily be calculated. Firstly, of course, $S(\xi)$ is given by $Q(\xi) + y^T My$. If it is assumed that the bound variables located by VEO4A are exact in the underlying model, then an estimate of the residual variance is given by $S(\xi)/(m-k)$. To estimate variances and covariances, an additional entry point is provided. This calculates $\{A\}^{-1}$ where $\{A\}$ indicates the submatrix $\{A_{ij}\}$ where i and j index only the free variables. The appropriate variance-covariance matrix for the free variables is then $\sigma^2 \{A\}^{-1}$. Estimates of standard deviations of the free variables are given by the square roots of the

diagonal elements of this matrix. Because the bound variables are known exactly, they have zero variance and covariance.

The entry point VEO4B is essentially written as a separate subroutine. It calculates $\{A\}^{-1}$ and is used as follows:

CALL VEO4B (N,A,IA,G,K)

N,A,IA,G and K must be passed on unchanged after exit from VEO4A.

VEO4B sets the following:

- A The off-diagonal elements of $\{A\}^{-1}$ are set in A(I,J) for $J < I \leq K$. The elements are indirectly addressed so that A(I,J) contains $\{A\}_{r,s}^{-1}$ where $r = LT(I)$ and $s = LT(J)$.
- C The diagonal elements of $\{A\}^{-1}$ are set in G(N+1), ..., G(N+K). They are again indirectly addressed so that G(N+I) contains $\{A\}_{r,r}^{-1}$ where $r = LT(I)$.

2nd May 1973

APPENDIX B
LISTING OF THE OPTIMIZATION CODE

```

SUBROUTINE VF01A(N,M,K,X,EPS,AKMIN,DFN,MAXFN,IPR1,IPR2,IW,MODE)
REAL X(1),EPS(1)
COMMON/VF01C/F,MM,KL,IS,MK,NU
COMMON/VF01D/G(50)
COMMON/VF01E/C(150)
COMMON/VF01F/GC(1250)
COMMON/VF01G/T(150)
COMMON/VF01H/GP(50)
COMMON/VF01I/G2P(325)
COMMON/VF01J/V(50)
COMMON/VF01K/WW(150)
COMMON/VF01L/W(2500)
COMMON/VF01M/ZZ(100)
COMMON/VF01N/LT(100)
COMMON/TIME/TMAX,TO,T1,IRS,MINS,AK,ITN,IFN
COMMON/INOM/INOM
EXTERNAL VF01Z
DATA BOUND/1.E+8/
1000 FORMAT(30I4)
1001 FORMAT(8E15.7)
NU=MAX0(25,N)
IF(M.GT.50)NU=N
IX=N
ICS=M
ICB=M+M
IS=M
IL=IS+M
IP=M
ILT=M
NN=N*(N+1)/2
MM=M
KL=K
R=1.
MK=0
INOM=1
CALL VF01B(N,M,X)
DF=DFN
IF(DFN.LT.0E0)DF=ABS(DFN*F)
IF(DFN.EQ.0E0)DF=F
IF(DF.LE.0.)DF=1.
IF(MODE.LT.0)GOTO5
DO 2 I=1,M
CC=C(I)
IF(I.GT.K)CC=A MIN1(CC,0E0)
IF(ABS(CC).GT.C(ICB+I))C(ICB+I)=ABS(CC)
2 CONTINUE
DO 3 I=1,M
T(IS+I)=2E0*DF/C(ICB+I)**2
3 T(I)=0.
5 CONTINUE

```

```

IF(IPR1.EQ.0)GOTO4
IF(MOD(MINS,IPR1).NE.0)GOTO4
PRINT 1002
1002 FORMAT('OENTRY TO VF01A'///'OCONSTRAINT SCALE PARAMETERS ARE')
PRINT 1001,(C(ICS+I),I=1,M)
4 CONTINUE
MD=IABS(MODE)
8 CONTINUE
DO 9 I=1,NN
9 W(I)=G2P(I)
IF(IPR1.EQ.0)GOTO7
IF(MOD(MINS,IPR1).NE.0)GOTO7
PRINT 1003,MINS
1003 FORMAT(////'OUTER ITERATION NUMBER IS',I3)
PRINT 1004
1004 FORMAT('OX(I)')
PRINT 1001,(X(I),I=1,N)
PRINT 1005
1005 FORMAT('OTHETA(I)')
PRINT 1001,(T(I),I=1,M)
PRINT 1006
1006 FORMAT('OSIGMA(I)')
PRINT 1001,(T(IS+I),I=1,M)
7 CONTINUE
IF(IRS.EQ.1) GO TO 82
ITN=0
IFN=0
82 IRS=0
CALL VA09A(VF01Z,N,X,PHI,GP,W,WW,DF,EPS,MD,MAXFN,IPR2,IEXIT)
IF(T1-T0.LT.TMAX) GO TO 80
DFN=DF
DO 81 I=1,NN
81 G2P(I)=W(I)
RETURN
80 CONTINUE
INOM=1
CALL VF01B(N,M,X)
MD=3
AKK=0.
DO 10 I=1,M
IF(I.GT.K.AND.C(I).EQ.0.) C(I)=1.E-8
CC=C(I)
IF(I.GT.K.AND.C(I).GE.T(I))CC=AMIN1(CC,0E0)
T(I)=T(I)*T(IS+I)
WW(I)=ABS(CC)/C(ICS+I)
IF(WW(I).GT.AKK)AKK=WW(I)
10 CONTINUE
IF(IPR1.EQ.0)GOTO16
IF(MOD(MINS,IPR1).NE.0)GOTO16
PRINT 1007

```

```

1007 FORMAT('OEXIT FROM VA09A'/'OLAGRANGE MULTIPLIERS USED IN VA09A')
PRINT 1001,(T(I),I=1,M)
PRINT 1008
1008 FORMAT('OLARGEST SCALED CONSTRAINT VIOLATION'/
1' THIS ITERATION, BEST ITERATION')
PRINT 1001,AKK,AK
PRINT 1009
1009 FORMAT('OCONSTRAINT RESIDUALS')
PRINT 1001,(C(I),I=1,M)
PRINT 1010
1010 FORMAT('OSCALED CONSTRAINT VIOLATIONS')
PRINT 1001,(WW(I),I=1,M)
16 CONTINUE
IF(IEXIT.EQ.0.OR.IEXIT.EQ.3)GOTO20
IF(AKK.LE.AKMIN)GOTO20
IF(AKK.GE.AK)GOTO11
DO 15 I=1,NN
15 G2P(I)=W(I)
DO 17 I=1,M
IF(I.GT.K.AND.T(I).EQ.0.0.AND.C(I).GT.0.0)GOTO17
ZZ(IP+I)=-T(IS+I)*C(I)
IF(I.GT.K.AND.ZZ(IP+I).LT.-T(I))ZZ(IP+I)=-T(I)
17 CONTINUE
IF(MINS.EQ.1)GOTO40
GOTO18
11 CONTINUE
DO 14 I=1,M
IF(WW(I).LE.AK.OR.C(ICB+I).GE.4E0*WW(I))GOTO14
DS=9E0*T(IS+I)
T(IS+I)=1E1*T(IS+I)
IF(IPR1.NE.0)PRINT 1011,I,T(IS+I)
1011 FORMAT('OSIGMA(,I3,)' INCREASED TO ',E15.7')
DO 12 J=1,N
12 V(J)=GC((I-1)*NU+J)
CALL MC11A(G2P,N,V,DS,V,N,N,DS)
14 CONTINUE
18 CONTINUE
DO 13 I=1,N
IF(ABS(X(I)-G(IX+I)).GT.EPS(I))GOTO21
13 CONTINUE
PRINT 1013
1013 FORMAT('OREQUESTED ACCURACY NOT OBTAINED')
20 CONTINUE
DO 2012 I=1,NN
2012 G2P(I)=W(I)
IF(IEXIT.EQ.0)PRINT 2000
2000 FORMAT('OMATRIX SET IN G2P BY USER IS NOT POSITIVE DEFINITE')
IF(IPR1.EQ.0)RETURN
PRINT 1012
1012 FORMAT('OBEST SOLUTION OBTAINED'/'--,(G(I),I=1,N)')


```

```

PRINT 1001,F,(G(I),I=1,N)
RETURN
21 CONTINUE
  IF(AKK.LT.AK)GOTO40
  DO 32 I=1,M
  32 V(I)=T(IL+I)
  GOTO70
40 CONTINUE
  MK=0
  KK=0
  DO 41 I=1,M
  T(IL+I)=T(I)
  C(ICB+I)=WW(I)
  IF(I.GT.K.AND.T(IL+I).EQ.0E0.AND.C(I).GT.0E0)GOTO41
  KK=KK+1
  LT(ILT+KK)=I
  GP(KK)=-BOUND
  IF(I.GT.K)GP(KK)=-T(IL+I)
  V(KK)=BOUND
  ZZ(KK)=-C(I)
41 CONTINUE
  IF(KK.EQ.0)GOTO20
  DO 42 I=1,N
42 G(IX+I)=X(I)
  KKK=KK*(KK+1)/2
  II=MAX0(KKK+NN,KK*KK)
  IF(II.LE.IW)GOTO50
  PRINT 2001,II
2001 FORMAT('0INCREASE STORAGE IN COMMON/VF01L TO',I7,'ELEMENTS')
  RETURN
50 CONTINUE
  II=IW-KKK
  DO 53 I=1,KK
  LI=LT(ILT+I)
  DO 51 JJ=1,N
51 X(JJ)=GC((LI-1)*NU+JJ)
  CALL MC11E(W,N,X,DUM1,X,N,1DUM,DUM2)
  DO 53 J=1,I
  LJ=LT(ILT+J)
  Z=0.
  DO 52 JJ=1,N
52 Z=Z+X(JJ)*GC((LJ-1)*NU+JJ)
  II=II+1
53 W(II)=Z
  JJ=IW-KKK
  II=0
  DO 56 I=1,KK
  DO 55 J=1,I
  JJ=JJ+1
55 W(II+J)=W(JJ)

```

```

56 II=II+KK
CALL VE04A(KK,W,KK,ZZ,GP,V,T,Z,LT,JJ,WW)
IF(IPR1.EQ.C)GOTO59
IF(MOD(MINS,IPR1).NE.0)GOTO59
PRINT 1020,KK
1020 FORMAT(I4,' ACTIVE CONSTRAINTS, NUMBERED')
PRINT 1000,(LT(ILT+I),I=1,KK)
PRINT 1021
1021 FORMAT('OLAGRANGE MULTIPLIER CORRECTIONS FOR ACTIVE CONSTRAINTS')
PRINT 1001,(T(I),I=1,KK)
59 CONTINUE
DO 60 I=1,M
60 V(I)=T(IL+I)
DO 62 I=1,KK
LI=LT(IL+I)
V(LI)=V(LI)+T(I)
Z=4E0*ABS((T(I)-ZZ(IP+LI))/ZZ(IP+LI))
IF(Z.LE.1E0)GOTO62
DS=(Z-1E0)*T(IS+LI)
T(IS+LI)=Z*T(IS+LI)
IF(IPR1.NE.0)PRINT 1011,LI,T(IS+LI)
DO 61 J=1,N
61 GP(J)=GC((LI-1)*NU+J)
CALL MC11A(G2P,N,GP,DS,GP,N,N,DS)
62 CONTINUE
AK=AKK
70 CONTINUE
DO 71 I=1,M
71 T(I)=V(I)/T(IS+I)
DO 72 I=1,N
72 X(I)=G(IX+I)
DF=1E50
MINS=MINS+1
GOTO8
END

```

```

SUBROUTINE VA09A(FUNCT,N,X,F,G,H,W,DFN,EPS,MODE,MAXFN,IPRINT,
1      IEXIT)
REAL X(1),G(1),H(1),W(1),EPS(1)
COMMON/TIME/TMAX,T0,T1,IRS,MINS,AK,ITN,IFN
COMMON/IGAMMA/IGAMMA
IF(IPRINT.NE.0)PRINT 1000
1000 FORMAT(//'ENTRY TO VA09A'|)
NN=N*(N+1)/2
IG=N
IGG=N+N
IS=IGG
IEXIT=0
IR=N
DF=DFN
IF(MODE.EQ.3)GOTO15
IF(MODE.EQ.2)GOTO10
IJ=NN+1
DO 5 I=1,N
DO 6 J=1,I
IJ=IJ-1
6 H(IJ)=0.
5 H(IJ)=1.
GOTO15
10 CONTINUE
CALL MC11B(H,N,D1,D2,D3,IH,ID1,D4)
IF(IR.LT.N)RETURN
15 CONTINUE
Z=F
CALL FUNCT(N,X,F,G)
IFN=IFN+1
IF(DFN.EQ.0.)DF=F-Z
IF(DFN.LT.0.)DF=ABS(DF*F)
IF(DF.LE.0.)DF=1.
20 CONTINUE
DFN=DF
CALL SECOND(T1)
IF(T1-T0.GE.TMAX)GO TO 90
IF(IPRINT.EQ.0)GOTO21
IF(MOD(ITN,IPRINT).NE.0)GOTO21
PRINT 1001,ITN,IFN
1001 FORMAT(24I5)
PRINT 1002,F
1002 FORMAT((8E15.7))
IF(IPRINT.LT.0)GOTO21
PRINT 1002,(X(I),I=1,N)
PRINT 1002,(G(I),I=1,N)
21 CONTINUE
ITN=ITN+1
DO 22 I=1,N
22 W(IG+I)=G(I)

```

```

CALL MC11E(H,N,G,D1,W,IR,ID1,D2)
GS=0.
DO 29 I=1,N
W(IS+I)=-G(I)
29 GS=GS-G(I)*W(IG+I)
IEXIT=2
IF(GS.GE.0.)GOTO92
GS0=GS
ALPHA=-2.*DF/GS
IF(ALPHA.GT.1.0) ALPHA=1.0
DF=F
TOT=0.
30 CONTINUE
IEXIT=3
IF(IFN.EQ.MAXFN)GOTO92
ICON=0
IEXIT=1
DO 31 I=1,N
Z=ALPHA*W(IS+I)
IF(ABS(Z).GE.EPS(I))ICON=1
31 X(I)=X(I)+Z
CALL FUNCT(N,X,FY,G)
IFN=IFN+1
IF(IGAMMA.EQ.0) GO TO 33
DO 34 I=1,N
Z=ALPHA*W(IS+I)
34 X(I)=X(I)-Z
ALPHA=0.1*ALPHA
PRINT 35,ALPHA
35 FORMAT(1X,"ALPHA=",F10.5)
GO TO 30
33 CONTINUE
GYS=0.
DO 32 I=1,N
32 GYS=GYS+G(I)*W(IS+I)
IF(FY.GE.F)GOTO40
IF(ABS(GYS/GS0).LE..9)GOTO50
IF(GYS.GT.0.)GOTO40
TOT=TOT+ALPHA
Z=10.
IF(GS.LT.GYS)Z=GYS/(GS-GYS)
IF(Z.GT.10.)Z=10.
ALPHA=ALPHA*Z
FY
GYS=GYS
GOTO30
40 CONTINUE
DO 41 I=1,N
41 X(I)=X(I)-ALPHA*W(IS+I)
IF(ICON.EQ.0)GOTO92

```

```

Z=3.* (F-FY)/ALPHA+GYS+GS
ZZ=SQRT(Z**2-GS*GYS)
Z=1.-(GYS+ZZ-Z)/(2.*ZZ+GYS-GS)
ALPHA=ALPHA*Z
GOTO30
50 CONTINUE
ALPHA=TOT+ALPHA
F=FY
IF(ICON.EQ.0)GOTO90
DF=DF-F
DGS=GYS-GS0
DO 51 I=1,N
W(IGG+I)=G(I)
51 G(I)=-W(IG+I)
IF(DGS+ALPHA*GS0.GT.0.)GOTO60
C  COMPLEMENTARY DFP FORMULA
SIG=1./GS0
IR=-IR
CALL MC11A(H,N,G,SIG,W,IR,1,0.)
DO 52 I=1,N
52 G(I)=W(IGG+I)-W(IG+I)
SIG=1./(ALPHA*DGS)
IR=-IR
CALL MC11A(H,N,G,SIG,W,IR,0,0.)
GOTO70
60 CONTINUE
C  DFP FORMULA
ZZ=ALPHA/(DGS-ALPHA*GS0)
SIG=-ZZ
CALL MC11A(H,N,G,SIG,W,IR,1,1E-13)
Z=DGS*ZZ-1.
DO 61 I=1,N
61 G(I)=W(IGG+I)+Z*W(IG+I)
SIG=1./(ZZ*DGS**2)
CALL MC11A(H,N,G,SIG,W,IR,0,0.)
70 CONTINUE
DO 71 I=1,N
71 G(I)=W(IGG+I)
GOTO20
92 CONTINUE
DO 91 I=1,N
91 G(I)=W(IG+I)
90 CONTINUE
IF(IPRINT.EQ.0)RETURN
PRINT 1001,ITN,IFN,IEXIT
PRINT 1002,F
PRINT 1002,(X(I),I=1,N)
PRINT 1002,(G(I),I=1,N)
RETURN
END

```

```
SUBROUTINE VF01Z(N,X,PHI,GPHI)
REAL X(1),GPHI(1)
COMMON/VF01C/F,M,K,IS,MK,NU
COMMON/VF01D/G(50)
COMMON/VF01E/C(150)
COMMON/VF01F/GC(1250)
COMMON/VF01G/T(150)
COMMON/IGAMMA/IGAMMA
IF(MK.EQ.1)CALL VF01B (N,M,X)
MK=1
IF(IGAMMA.EQ.1) RETURN
PHI=0.
DO 10 I=1,N
10 GPHI(I)=G(I)
DO 12 I=1,M
CC=C(I)-T(I)
IF(I.GT.K)CC=AMIN1(CC,OE0)
Y=T(IS+I)*CC
IF(Y.EQ.OE0)GOTO12
PHI=PHI+Y*CC
DO 11 J=1,N
11 GPHI(J)=GPHI(J)+Y*GC((I-1)*NU+J)
12 CONTINUE
PHI=.5E0*PHI+F
RETURN
END
```

```

SUBROUTINE MC11A(A,N,Z,SIG,W,IR,MK,EPS)
DIMENSION A(1),Z(1),W(1)
C UPDATE FACTORS GIVEN IN A BY SIG*Z*ZTRANSPOSE
IF(N.GT.1)GOTO1
IR=1
A(1)=A(1)+SIG *2(1)**2
IF(A(1).GT.0.)RETURN
A(1)=0.
IR=0
RETURN
1 CONTINUE
NP=N+1
IF(SIG.GT.0.)GOTO40
IF(SIG.EQ.0..OR.IR.EQ.0)RETURN
TI=1./SIG
IJ=1
IF(MK.EQ.0.)GOTO10
DO 7 I=1,N
IF(A(IJ).NE.0.)TI=TI+W(I)**2/A(IJ)
7 IJ=IJ+NP-1
GOTO20
10 CONTINUE
DO 11 I=1,N
11 W(I)=Z(I)
DO 15 I=1,N
IP=I+1
V=W(I)
IF(A(IJ).GT.0.)GOTO12
W(I)=0.
IJ=IJ+NP-1
GOTO15
12 CONTINUE
TI=TI+V**2/A(IJ)
IF(I.EQ.N)GOTO14
DO 13 J=IP,N
IJ=IJ+1
13 W(J)=W(J)-V*A(IJ)
14 IJ=IJ+1
15 CONTINUE
20 CONTINUE
IF(IR.LE.0 )GOTO21
IF(TI.GT.0.)GOTO22
IF(MK-1)40,40,23
21 TI=0.
IR=-IR-1
GOTO23
22 TI=EPS/SIG
IF(EPS.EQ.0.)IR=IR-1
23 CONTINUE
MM=1

```

```

TIM=TI
DO 30 I=1,N
J=NP-I
IJ=IJ-I
IF(A(IJ).NE.0.)TIM=TI-W(J)**2/A(IJ)
W(J)=TI
30 TI=TIM
GOTO41
40 CONTINUE
MM=0
TIM=1./SIG
41 CONTINUE
IJ=1
DO 66 I=1,N
IP=I+1
V=Z(I)
IF(A(IJ).GT.0.)GOTO53
IF(IR.GT.0 .OR.SIG.LT.0..OR.V.EQ.0.)GOTO52
IR=1-IR
A(IJ)=V**2/TIM
IF(I.EQ.N)RETURN
DO 51 J=IP,N
IJ=IJ+1
51 A(IJ)=Z(J)/V
RETURN
52 CONTINUE
TI=TIM
IJ=IJ+NP-I
GOTO66
53 CONTINUE
AL=V/A(IJ)
IF(MM)54,54,55
54 TI=TIM+V*AL
GOTO56
55 TI=W(I)
56 CONTINUE
R=TI/TIM
A(IJ)=A(IJ)*R
IF(R.EQ.0.)GOTO70
IF(I.EQ.N)GOTO70
B=AL/TI
IF(R.GT.4.)GOTO62
DO 61 J=IP,N
IJ=IJ+1
Z(J)=Z(J)-V*A(IJ)
61 A(IJ)=A(IJ)+B*Z(J)
GOTO64
62 GM=TIM/TI
DO 63 J=IP,N
IJ=IJ+1

```

```

Y=A(IJ)
A(IJ)=B*Z(J)+Y*GM
63 Z(J)=Z(J)-V*Y
64 CONTINUE
  TIM=TI
  IJ=IJ+1
66 CONTINUE
70 CONTINUE
  IF(IR.LT.0)IR=-IR
  RETURN
C  FACTORIZE A MATRIX GIVEN IN A
  ENTRY MC11B
  IR=N
  IF(N.GT.1)GOTO100
  IF(A(1).GT.0.)RETURN
  A(1)=0.
  IR=0
  RETURN
100 CONTINUE
  NP=N+1
  II=1
  DO 104 I=2,N
    AA=A(II)
    NI=II+NP-I
    IF(AA.GT.0.)GOTO101
    A(II)=0.
    IR=IR-1
    II=NI+1
    GOTO104
101 CONTINUE
  IP=II+1
  II=NI+1
  JK=II
  DO 103 IJ=IP,NI
    V=A(IJ)/AA
    DO 102 IK=IJ,NI
      A(JK)=A(JK)-A(IK)*V
102 JK=JK+1
103 A(IJ)=V
104 CONTINUE
  IF(A(II).GT.0.)RETURN
  A(II)=0.
  IR=IR-1
  RETURN
C  MULTIPLY OUT THE FACTORS GIVEN IN A
  ENTRY MC11C
  IF(N.EQ.1)RETURN
  NP=N+1
  II=N*NP/2
  DO 202 NIP=2,N

```

```

JK=II
NI=II-1
II=II-NIP
AA=A(II)
IP=II+1
IF(AA.GT.0.)GOTO203
DO 204 IJ=IP,NI
204 A(IJ)=0.
GOTO202
203 CONTINUE
DO 201 IJ=IP,NI
V=A(IJ)*AA
DO 200 IK=IJ,NI
A(JK)=A(JK)+A(IK)*V
200 JK=JK+1
201 A(IJ)=V
202 CONTINUE
RETURN
C  MULTIPLY A VECTOR Z BY THE FACTORS GIVEN IN A
ENTRY MC11D
IF(N.GT.1)GOTO300
Z(1)=Z(1)*A(1)
W(1)=Z(1)
RETURN
300 CONTINUE
NP=N+1
II=1
N1=N-1
DO 303 I=1,N1
Y=Z(I)
IF(A(II).EQ.0.)GOTO302
IJ=II
IP=I+1
DO 301 J=IP,N
IJ=IJ+1
301 Y=Y+Z(J)*A(IJ)
302 Z(I)=Y*A(II)
W(I)=Z(I)
303 II=II+NP-I
Z(N)=Z(N)*A(II)
W(N)=Z(N)
DO 311 K=1,N1
I=N-K
II=II-NP+I
IF(Z(I).EQ.0.)GOTO311
IP=I+1
IJ=II
Y=Z(I)
DO 310 J=IP,N
IJ=IJ+1

```

```

310 Z(J)=Z(J)+A(IJ)*Z(I)
311 CONTINUE
    RETURN
C   MULTIPLY A VECTOR Z BY THE INVERSE OF THE FACTORS GIVEN IN A
    ENTRY MC11E
    IF(IR.LT.N)RETURN
    W(1)=Z(1)
    IF(N.GT.1)GOTO400
    Z(1)=Z(1)/A(1)
    RETURN
400 CONTINUE
    DO 402 I=2,N
    IJ=I
    I1=I-1
    V=Z(I)
    DO 401 J=1,I1
    V=V-A(IJ)*Z(J)
401 IJ=IJ+N-J
    W(I)=V
402 Z(I)=V
    Z(N)=Z(N)/A(IJ)
    NP=N+1
    DO 411 NIP=2,N
    I=NP-NIP
    II=IJ-NIP
    V=Z(I)/A(II)
    IP=I+1
    IJ=II
    DO 410 J=IP,N
    II=II+1
410 V=V-A(II)*Z(J)
411 Z(I)=V
    RETURN
C   COMPUTE THE INVERSE MATRIX FROM FACTORS GIVEN IN A
    ENTRY MC11F
    IF(IR.LT.N)RETURN
    A(1)=1./A(1)
    IF(N.EQ.1)RETURN
    NP=N+1
    N1=N-1
    II=2
    DO 511 I=2,N
    A(II)=-A(II)
    IJ=II+1
    IF(I.EQ.N)GOTO502
    DO 501 J=I,N1
    IK=II
    JK=IJ
    V=A(IJ)
    DO 500 K=I,J

```

```
JK=JK+NP-K
V=V+A(IK)*A(JK)
500 IK=IK+1
A(IJ)=V
501 IJ=IJ+1
502 CONTINUE
A(IJ)=1./A(IJ)
IJ=IJ+1
AA=A(IJ)
IJ=I
IP=I+1
NI=N-I
DO 511 J=2,I
V=A(IJ)*AA
IK=IJ
K=IJ-IP+J
I1=IJ-1
NIP=NI+I,J
DO 510 JK=K,I1
A(JK)=A(JK)+V*A(IK)
510 IK=IK+NIP-JK
A(IJ)=V
511 IJ=IJ+NP-J
RETURN
END
```

```

SUBROUTINE VE04A(N,A,IA,B,BL,BU,X,Q,LT,K,G)
DIMENSION A(IA,1),B(1),BL(1),BU(1),X(1),LT(1),G(1)
IS=N
IAS=N
IV=N
ICAC=N+N
ID=ICAC
DO 9 I=1,N
9 G(I)=-B(I)
DO 10 I=1,N
X(I)=0.
LT(I)=I
G(ICAC+I)=A(I,I)
IF(0..GE.BL(I).AND.0..LE.BU(I))GOTO10
IF(0..LT.BL(I))X(I)=BL(I)
IF(0..GT.BU(I))X(I)=BU(I)
DO 12 J=1,I
12 G(J)=G(J)+A(J,I)*X(I)
IF(I.EQ.N)GOTO10
II=I+1
DO 11 J=II,N
11 G(J)=G(J)+A(I,J)*X(I)
10 CONTINUE
K=0
K1=1
20 CONTINUE
IOUT=0
DEL=0.
DO 21 I=K1,N
LI=LT(I)
IF(X(LI).EQ.BL(LI).AND.G(I).GE.0.)GOTO21
IF(X(LI).EQ.BU(LI).AND.G(I).LE.0.)GOTO21
IF(G(I).LT.0.)GOTO22
Z=X(LI)-BL(LI)
J=1
GOTO23
22 CONTINUE
Z=BU(LI)-X(LI)
J=0
23 CONTINUE
IF(G(ICAC+I).LE.0.)GOTO24
BETA=ABS(G(I))/G(ICAC+I)
IF(BETA.GE.Z)GOTO24
Z=BETA
D=.5*Z*ABS(G(I))
J=-1
GOTO26
24 CONTINUE
D=Z*(ABS(G(I))-Z*G(ICAC+I))
26 CONTINUE

```

```

IF(D.LT.DEL)GOTO21
DEL=D
ALPHA=Z
IOUT=I
IIN=I
IF(J.LT.0)IIN=0
LB=J
21 CONTINUE
IF(IOUT.NE.0)GOTO29
27 CONTINUE
Q=0.
DO 28 I=1,N
LI=LT(I)
28 Q=Q+X(LI)*(G(I)-B(LI))
Q=.5*Q
RETURN
29 CONTINUE
SIG=1.
IF(G(IOUT).GT.0.)SIG=-1.
LIOUT=LT(IOUT)
LIIN=LIOUT
25 CONTINUE
SAS=G(ICAC+IOUT)
IF(K.EQ.0)GOTO31
DO 30 I=1,K
30 G(IS+I)=G(ID+I)*A(IOUT,I)
31 CONTINUE
DO 37 I=K1,N
LI=LT(I)
IF(LI-LIOUT)32,37,33
32 Z=A(LI,LIOUT)
GOTO34
33 Z=A(LIOUT,LI)
34 CONTINUE
IF(K.EQ.0)GOTO36
DO 35 J=1,K
35 Z=Z-A(I,J)*G(IS+J)
36 G(IS+I)=Z
37 CONTINUE
G(IS+IOUT)=SAS
IF(K.EQ.0)GOTO42
G(IS+K)=-A(IOUT,K)
IF(K.EQ.1)GOTO42
I=K
DO 41 II=2,K
I=I-1
Z=-A(IOUT,I)
I1=I+1
DO 40 J=I1,K
40 Z=Z-G(IS+J)*A(J,I)

```

```

41 G(IS+I)=Z
42 CONTINUE
  IF(SIG.EQ.1.)GOTO51
  DO 50 I=1,N
50 G(IS+I)=-G(IS+I)
51 CONTINUE
  IF(K.EQ.0)GOTO62
  DO 61 I=1,K
  IF(G(IS+I).EQ.0.)GOTO61
  LI=LT(I)
  J=1
  Z=BL(LI)-X(LI)
  IF(G(IS+I).LT.0.)GOTO60
  J=0
  Z=BU(LI)-X(LI)
60 CONTINUE
  Z=Z/G(IS+I)
  IF(Z.GE.ALPHA)GOTO61
  ALPHA=Z
  LB=J
  IIN=I
  LIIN=LI
61 CONTINUE
62 CONTINUE
  X(LIOUT)=X(LIOUT)+SIG*ALPHA
  IF(K.EQ.0)GOTO71
  DO 70 I=1,K
  LI=LT(I)
70 X(LI)=X(LI)+ALPHA*G(IS+I)
71 CONTINUE
  DO 72 I=K1,N
72 G(I)=G(I)+ALPHA*G(IAS+I)
  IF(IIN.EQ.0)GOTO90
  X(LIIN)=BL(LIIN)
  IF(LB.EQ.0)X(LIIN)=BU(LIIN)
  IF(IIN.EQ.IOUT)GOTO20
  K2=K-1
  SG=G(ID+IIN)
  I1=IIN+1
  DO 80 I=I1,N
80 G(IV+I)=A(I,IIN)
  IF(IIN.EQ.K)GOTO86
  I2=IIN+2
  S0=1./SG
  DO 85 I=IIN,K2
  V=G(IV+I1)
  VD=V/G(ID+I1)
  S1=S0+V*VD
  R=S1/S0
  G(ID+I)=G(ID+I1)*R

```

```

BETA=VD/S1
IF(R.GT.4.)GOTO841
DO 81 J=I2,N
81 G(IV+J)=G(IV+J)-V*A(J,I1)
IF(I1.GT.K2)GOTO83
DO 82 J=I1,K2
J1=J+1
82 A(J,I)=A(J1,I1)+BETA*G(IV+J1)
83 CONTINUE
A(K,I)=BETA
DO 84 J=K1,N
84 A(J,I)=A(J,I1)+BETA*G(IV+J)
GOTO849
841 CONTINUE
IF(I1.GT.K2)GOTO843
DO 842 J=I1,K2
J1=J+1
842 A(J,I)=BETA*G(IV+J1)+A(J1,I1)/R
843 CONTINUE
A(K,I)=BETA
DO 844 J=K1,N
844 A(J,I)=BETA*G(IV+J)+A(J,I1)/R
DO 845 J=I2,N
845 G(IV+J)=G(IV+J)-V*A(J,I1)
849 CONTINUE
LT(I)=LT(I1)
S0=S1
I1=I2
85 I2=I2+1
SG=1./S1
LT(K)=LIIN
G(ID+K)=SG
IF(IIN.EQ.1)GOTO851
II=IIN-1
DO 852 I=1,II
Z=A(IIN,I)
DO 853 J=IIN,K2
853 A(J,I)=A(J+1,I)
852 A(K,I)=Z
851 CONTINUE
86 CONTINUE
DO 87 I=K1,N
87 G(ICAC+I)=G(ICAC+I)+SG*G(IV+I)**2
K1=K
K=K2
IIN=0
ALPHA=1E75
SAS=G(ICAC+IOUT)
IF(SAS.GT.0.)ALPHA=ABS(G(IOUT))/SAS
IF(G(IOUT).LT.0.)GOTO898

```

```

J=1
Z=X(LIOUT)-BL(LIOUT)
GOTO899
898 CONTINUE
J=0
Z=BU(LIOUT)-X(LIOUT)
899 CONTINUE
IF(Z.GE.ALPHA)GOTO25
ALPHA=Z
LB=J
IIN=IOUT
LIIN=LIOUT
GOTO25
90 CONTINUE
K2=K1+1
IF(SIG.EQ.1.)GOTO91
DO 901 I=K1,N
901 G(IAS+I)=-G(IAS+I)
91 CONTINUE
IF(IOUT.EQ.K1)GOTO97
LT(IOUT)=LT(K1)
LT(K1)=LIOUT
G(IAS+IOUT)=G(IAS+K1)
G(ICAC+IOUT)=G(ICAC+K1)
G(ICAC+K1)=SAS
G(IOUT)=G(K1)
IF(K.EQ.0)GOTO97
DO 92 I=1,K
Z=A(K1,I)
A(K1,I)=A(IOUT,I)
92 A(IOUT,I)=Z
93 CONTINUE
IF(K2.EQ.IOUT)GOTO95
I1=IOUT-1
DO 94 I=K2,I1
94 A(IOUT,I)=A(I,K1)
95 CONTINUE
IF(IOUT.EQ.N)GOTO97
I1=IOUT+1
DO 96 I=I1,N
96 A(I,IOUT)=A(I,K1)
97 CONTINUE
G(K1)=0.
K=K1
IF(K.EQ.N)GOTO27
DO 98 I=K2,N
Z=G(IAS+I)/SAS
A(I,K1)=Z
98 G(ICAC+I)=G(ICAC+I)-Z*G(IAS+I)
K1=K2

```

```

GOTO20
C
ENTRY VEN 3
IF(K.EQ.0)RETURN
ID=N+N
G(N+1)=1./G(ID+1)
IF(K.EQ.1)RETURN
N1=K-1
DO 111 I=1,N1
I1=I+1
A(I1,I)=-A(I1,I)
IF(I.EQ.N1)GOTO102
II=I+2
DO 101 J=II,K
Z=A(J,I)
J1=J-1
DO 100 L=I1,J1
100 Z=Z+A(J,L)*A(L,I)
101 A(J,I)=-Z
102 CONTINUE
AA=1./G(ID+I1)
G(N+I1)=AA
DO 111 J=1,I
Z=A(I1,J)*AA
G(N+J)=G(N+J)+Z*A(I1,J)
IF(I.EQ.1)GOTO111
J1=J+1
DO 110 L=J1,I
110 A(L,J)=A(L,J)+A(I1,L)*Z
111 A(I1,J)=Z
RETURN
END

```

APPENDIX C
LISTING OF THE USER CODE

```

PROGRAM MRRV(INPUT,OUTPUT,TAPE10)
DIMENSION X(16),EPS(16)
COMMON/NFEST/NFEST
COMMON/VF01E/C(150)
COMMON/VF01F/GC(25,50)
COMMON/VF01G/T(150)
COMMON/VF01I/G2P(325)
COMMON/P/TF,NA,TTA(7),TA(7),NM,TTM(7),TM(7)
COMMON/CNSTR/ME,MI,MC
COMMON/TIME/TMAX,TO,T1,IRS,MINS,AK,ITN,IFN
COMMON/INOM/INOM
COMMON/IPROB/IPROB
DATA DPR/57.29577951/
IPROB=2
IPROB=1
IRS=1
IRS=0
TMAX=1200.
N=11
NA=5
NM=5
M=1
MC=M
K=1
ME=K
M1=M-K
X(1)=3200./806.48263
DO 5 I=1,5
TTA(I)=FLOAT(I-1)*0.25
X(I+1)=20./DPR
TTM(I)=FLOAT(I-1)*0.25
X(I+NA+1)=60./DPR
5 CONTINUE
DO 1 I=1,N
EPS(I)=1.E-4*X(I)
1 CONTINUE
AKMIN=1.E-4
DFN=.5
MAXFN=10000
IPR1=1
IPR2=1
IW=2500
MODE=1
C(M+1)=0.136
AK=1E60
M2=M+M
NN=N*(N+1)/2
MINS=1
ITN=0
IFN=0

```

```

      PRINT 20
20 FORMAT(1H1)
      IF(IRS.EQ.0) GO TO 8
      READ(10,16) IRS,MINS,MODE,NFEST,ITN,IFN,(X(I),I=1,N),(C(I),I=1,M2
      1 ),(T(I),I=1,M2),(G2P(I),I=1,NN),AK,DFN
16 FORMAT(9X,6(I10),135(/,9X,5020))
      PRINT 17,    IRS,MINS,MODE,NFEST,ITN,IFN,(X(I),I=1,N),(C(I),I=1,M2
      1 ),(T(I),I=1,M2),(G2P(I),I=1,NN),AK,DFN
17 FORMAT(9X,6(I10),135(/,9X,5E24.14))
      PRINT 15
15 FORMAT(//1X*INPUT VALUES OF THETA AND SIGMA USED*)
      PRINT 22,NFEST
22 FORMAT(1X*INPUT VALUE OF HESSIAN USED*//1X*    TOTAL FUNCTION *
*$*EVALUATIONS SO FAR*I8)
8 CONTINUE
      CALL SECOND(TO)
      CALL VF01A(N,M,K,X,EPS,AKMIN,DFN,MAXFN,IPR1,IPR2,IW,MODE)
      IF(T1-TO.LT.TMAX) GO TO 9
      IRS=1
      PRINT 30,TMAX
30 FORMAT(//1X*.....TIME LIMIT OF*G15.8*EXCEEDED.....*//)
      PRINT 31,T1-TO
31 FORMAT(1X*COMPUTATION TIME*G16.8/)
      MODE=-3
9 REWIND 10
      WRITE(10,16) IRS,MINS,MODE,NFEST,ITN,IFN,(X(I),I=1,N),(C(I),I=1,M2
      1 ),(T(I),I=1,M2),(G2P(I),I=1,NN),AK,DFN
      PRINT 17,    IRS,MINS,MODE,NFEST,ITN,IFN,(X(I),I=1,N),(C(I),I=1,M2
      1 ),(T(I),I=1,M2),(G2P(I),I=1,NN),AK,DFN
      PRINT 10,NFEST
10 FORMAT(//5X*TOTAL FUNCTION EVALUATIONS*1XI6)
      END

```

```
SUBROUTINE VF01B(N,M,X)
DIMENSION X(1)
COMMON/NFEST/NFEST
COMMON/VF01C/F,MM,K,IS,MMK,NU
COMMON/VF01D/G(50)
COMMON/VF01E/C(150)
COMMON/VF01F/GC(25,50)
COMMON/IGAMMA/IGAMMA
DATA NFEST/0/
CALL SG(X,F)
IF(IGAMMA.EQ.1) RETURN
NFEST=NFEST+1
CALL SGX(N,X,M,NFEST)
RETURN $ END
```

```

SUBROUTINE SG(X,F)
REAL M,N,MASS
DIMENSION X(16),Y(9),Z(9),DY(9)
COMMON/VFO1E/S(150)
COMMON/SS/SS(50)
COMMON/INOM/INOM
COMMON/P/TF,NA,TTA(7),TA(7),NM,TTM(7),TM(7)
COMMON/CNSTR/ME,MI,MC
COMMON/IGX/IGX
COMMON/TRAJ/A,M,N,KOUNT,TIMETB,MASS,NDE
COMMON/COSI/CF,VCGOR,CSI,SSI
COMMON/IPROB/IPROB
COMMON/IGAMMA/IGAMMA
DATA W,IGX,DPR/5.8809641E-2,0.57.29577951/
IGAMMA=0
TF=X(1)
DO 5 I=1,NA
5 TA(I)=X(I+1)
DO 6 I=1,NM
6 TM(I)=X(I+NA+1)
TIMETB=X(1+NA+NM+1)
IF(IPROB.EQ.2) GO TO 59
TIMETB=10.
59 CONTINUE
NDE=6
DELT=.004 $ NIS=250
T=0.
Y(1)=0.0 $ Y(2)=0.0 $ Y(3)=1.017432502
Y(4)=.94420438 $ Y(5)=-.020943951 $ Y(6)=0.
Y(7)=0. $ Y(8)=0.
KOUNT=0
TIMETL=TIMETB
DO 15 I=1,NIS
IF(INOM.NE.1) GO TO 25
IF(MOD(I,25).NE.1) GO TO 25
IF(I.EQ.1) PRINT 60
60 FORMAT(//,2X,*TAU*,4X,*TIME*,4X,*THETA*,4X,*PHI*,2X,*ALTITUDE*,1X,
1*VELOCITY*,2X,*GAMMA*,4X,*PSI*,4X,*ALPHA*,5X,*MU*,,
26X,*AP*,6X,*MP*,6X,*N*,/,,
39X,*(SEC)*,3X,* (DEG)*,3X,* (DEG)*,3X,* (FT)*,3X,* (FT/SEC)*,2X,
4 * (DEG)*,3X,* (DEG)*,3X,* (DEG)*,3X,* (DEG)*)
CALL DERIV(T,Y,DY)
Z(1)=DPR*Y(1) $ Z(2)=DPR*Y(2) $ Z(3)=2.0926428E7*(Y(3)-1.)
Z(4)=25947.772*Y(4) $ Z(5)=DPR*Y(5) $ Z(6)=DPR*Y(6)
Z(7)=806.48263*Y(7) $ Z(8)=806.48263*Y(8)
PA=DPR*A $ PM=DPR*M
TIME=806.48263*T*TF
TEMP=VCGOR*CSI+W*CF
CI=CF*TEMP/SQRT(TEMP**2+(VCGOR*SSI)**2)
PRINT 20,T,TIME,(Z(J),J=1,6),PA,PM,Z(7),Z(8),N

```

```

20 FORMAT(1X,F5.3,2X,F6.1,2X,F6.2,2X,F6.3,2X,F7.0,2X,F6.0,2X,F6.2,2X,
1   F6.2,2X,F6.3,2X,F6.2,2X,F6.3,2X,F6.3,2X,F6.3)
25 CONTINUE
   IF(KOUNT.EQ.2) GO TO 10
   DELT1=TIMETL/TF-T
   IF(DELT1.LE.DELT) GO TO 11
10 CALL RUNGE(T,Y,DELT,NDE)
   IF(Y(5).GT.-1.) GO TO 50
   IGAMMA=1
   RETURN
50 CONTINUE
   GO TO 15
11 CONTINUE
   CALL RUNGE(T,Y,DELT1,NDE)
   KOUNT=KOUNT+1
   IF(INOM.NE.1) GO TO 63
   TIMEL=806.48263*TIMETL
   IF(KOUNT.EQ.1) PRINT 61,TIMEL
61 FORMAT(1H ,*IGNITION OCCURRED AT *,F10.5,* SECONDS*)
63 CONTINUE
   IF(DELT1.EQ.DELT) GO TO 12
   DELT2=DELT-DELT1
   CALL RUNGE(T,Y,DELT2,NDE)
12 TIMETL=TIMETL+0.63901697
15 CONTINUE
   CALL DERIV(T,Y,DY)
   TEMP=VCGOR*CSI+W*CF
   CI=CF*TEMP/SQRT(TEMP**2+(VCGOR*SSI)**2)
   IF(INOM.NE.1) GO TO 26
   Z(1)=DPR*Y(1) $ Z(2)=DPR*Y(2) $ Z(3)=2.0926428E7*(Y(3)-1.)
   Z(4)=25947.772*Y(4) $ Z(5)=DPR*Y(5) $ Z(6)=DPR*Y(6)
   Z(7)=806.48263*Y(7) $ Z(8)=806.48263*Y(8)
   PA=DPR*A $ PM=DPR*M
   TIME=806.48263*T*TF
   PRINT 20,T,TIME,(Z(J),J=1,6),PA,PM,Z(7),Z(8),N
26 CONTINUE
   INOM=0
   IF(IPROB.EQ.2) GO TO 72
   F=-Y(2)
   SS(1)=(Y(3)-1.0047786464)/0.0047786464
   SS(2)=Y(7)
   SS(3)=Y(8)
   GO TO 74
72 CONTINUE
   F=CI
   SS(1)=(Y(3)-1.029054170)/.029054170
   SS(2)=Y(4)/.90566542-1.
   SS(3)=Y(5)
   SS(4)=Y(7)
   SS(5)=Y(8)

```

```
74 CONTINUE
  IF(IGX.EQ.1) GO TO 40
  DO 35 I=1,MC
    S(I)=SS(I)
35 CONTINUE
  PRINT 28,F,(S(J),J=1,MC)
28 FORMAT(1X,/,1X,6F20.15,/)

40 CONTINUE
  RETURN
END
```

```
SUBROUTINE RUNGE (T,X,DELT,N)
DIMENSION X(10),DX(10),DELX(10,3),XV(10)
CALL DERIV (T,X,DX)
T2 = T + DELT/2.0
DO 100 I = 1,N
  DELX(I,1) = DX(I)*DELT
100 XV(I) = X(I) + DELX(I,1)/2.0
  CALL DERIV (T2,XV,DX)
  DO 200 I=1,N
    DELX(I,2) = DX(I) *DELT
200 XV(I) = X(I) + DELX(I,2)/2.0
  CALL DERIV (T2,XV,DX)
  DO 300 I=1,N
    DELX(I,3) = DX(I)*DELT
300 XV(I) = X(I) + DELX(I,3)
  T = T + DELT
  CALL DERIV (T,XV,DX)
  DO 400 I=1,N
400 X(I) = X(I) + (DELX(I,1) + DX(I)*DELT + 2.0*(DELX(I,2) + DELX(I,3)
1))/6.0
  RETURN
END
```

```

SUBROUTINE DERIV(TT,Y,DY)
REAL L,M,N,NLIM,MASS
DIMENSION Y(9),DY(9)
COMMON/P/TTF,NA,TTA(7),TA(7),NM,TTM(7),TM(7)
COMMON/TRAJ/A,M,N,KOUNT,TIMETB,MASS,NDE
COMMON/COSI/CF,VCGOR,CS,SS
DATA TW,W2,ALIM,NLIM/.11761928,3.4585739E-9,.69813170,4.5/
T=Y(1) $ F=Y(2) $ R=Y(3) $ V=Y(4) $ G=Y(5) $ S=Y(6)
SF=SIN(F) $ CF=COS(F)
SG=SIN(G) $ CG=COS(G)
SS=SIN(S) $ CS=COS(S)
TF=SF/CF $ TG=SG/CG
CALL SLIN1(TT,A,TTA,TA,NA)
CALL SLIN1(TT,M,TTM,TM,NM)
SA=SIN(A) $ CA=COS(A) $ SM=SIN(M) $ CM=COS(M)
VCG=V*CG
VCGOR=VCG/R
CALL AERO(R,V,A,SA,CA,D,L)
GR=1./R**2
IF(KOUNT.GT.0) GO TO 5
MASS=1.
THR=0.
GO TO 15
5 IF(KOUNT.GT.1) GO TO 10
MASS=1.-.83533982*(TT*TTF-TIMETB)
THR=.30555556
GO TO 15
10 MASS=.46620367
THR=0.
15 CONTINUE
N=L/MASS
TMDOM=(THR*CA-D)/MASS
TPLOM=(THR*SA+L)/MASS
TDOT=VCGOR*CS/CF
FDOT=VCGOR*SS
RDOT=V*SG
VDOT=TMDOM-GR*SG+W2*R*CF*(SG*CF-CG*SF*SS)
GDOT=TPLOM*CM/V-GR*CG/V+VCGOR+TW*CF*CS+W2*(R*CF/V)*(CG*CF+SG*SF*SS
1  )
SDOT=TPLOM*SM/VCG-VCGOR*TF*CS+TW*(TG*SS*CF-SF)-(W2/VCGOR)*SF*CF*CS
DY(1)=TDOT*TTF $ DY(2)=FDOT*TTF $ DY(3)=RDOT*TTF
DY(4)=VDOT*TTF $ DY(5)=GDOT*TTF $ DY(6)=SDOT*TTF
IF(NDE.EQ.6) GO TO 20
P=1.-A/ALIM
IF(P.GT.0.) P=0.
DY(7)=-P*P
P=M
IF(P.GT.0.) P=0.
DY(8)=-P*P
DY(7)=DY(7)*TTF $ DY(8)=DY(8)*TTF
20 CONTINUE
RETURN $ END

```

```

SUBROUTINE AERO(R,V,AL,SAL,CAL,D,L)
REAL L,M
DIMENSION TM(5),TAA(5),TBB(5),TCC(5),TA1(5),TA2(5),TB1(5),TB2(5),T
AC1(5),TC2(5),TD1(5),TD2(5),TD3(5)
COMMON/IPROB/IPROB
DATA TM/.2,1,2,5,.10.,.20./
DATA TAA/0.,0.,0.,2.4E-12,1.38E-12/
DATA TBB/7.8E-8,7.7E-8,5.6E-8,-7.11E-7,-3.81E-7/
DATA TCC/.0114,.0076,.0028,.0578,.0297/
DATA TA1/-1.0E-4,-7.81E-5,1.09E-5,1.41E-5,1.33E-5/
DATA TB1/-2.19E-3,-1.25E-3,-9.62E-4,-9.0E-4,-8.19E-4/
DATA TA2/-1.0E-4,-7.81E-5,4.86E-6,-6.6E-6,-6.25E-6/
DATA TB2/-2.19E-3,-1.25E-3,-1.13E-4,3.49E-4,3.58E-4/
DATA TC1/.035,.076,.0324,.0261,.0243/
DATA TC2/.035,.076,.0204,.0114,.0105/
DATA TD1/1.33E-4,-7.81E-5,2.94E-4,4.38E-4,4.38E-4/
DATA TD2/2.68E-2,2.90E-2,1.41E-2,6.50E-3,6.50E-3/
DATA TD3/-0.030,-0.030,-0.035,-0.035,-0.035/
DATA DPR,S/57.29577951,125.84/
H=20926428.* (R-1.)
VEL=25947.772*V
ALFA=DPR*AL
CALL ATM62(H,TAU,SIGMA)
M=VEL/(65.770*SQRT(TAU))
CALL SLIN1(M,AA,TM,TAA,5)
CALL SLIN1(M,BB,TM,TBB,5)
CALL SLIN1(M,CC,TM,TCC,5)
IF(ALFA.GT.16.)GO TO 10
CALL SLIN1(M,A,TM,TA1,5)
CALL SLIN1(M,B,TM,TB1,5)
CALL SLIN1(M,C,TM,TC1,5)
GO TO 20
10 CALL SLIN1(M,A,TM,TA2,5)
CALL SLIN1(M,B,TM,TB2,5)
CALL SLIN1(M,C,TM,TC2,5)
20 CONTINUE
CALL SLIN1(M,D1,TM,TD1,5)
CALL SLIN1(M,D2,TM,TD2,5)
CALL SLIN1(M,D3,TM,TD3,5)
CASF=AA*H*H+BB*H+CC
CAPR=A*ALFA*ALFA+B*ALFA+C
CA=CASF+CAPR
CN=D1*ALFA*ALFA+D2*ALFA+D3
CL=CN*SAL-CA*SAL
CD=CA*SAL+CN*SAL
IF(IPROB.EQ.2) GO TO 30
BB=15268.635*SIGMA*V**2
GO TO 40
30 BB=932.34367*SIGMA*V**2
40 CONTINUE
D=BB*CD
L=BB*CL
RETURN
END

```

```
SUBROUTINE ATM62(H,T,S)
REAL LM(8)
DIMENSION HM(8),TM(8),CS(8)
DATA HM/0.,36089.,65617.,104987.,154199.,170604.,200131.,259186./
DATA TM/288.15,216.65,216.65,228.65,270.65,270.65,252.65,180.65/
DATA LM/-1.9812E-03,0.,+3.048E-04,8.5344E-04,0.,-6.096E-04,
1 -1.2192E-03,0./
DATA CS/3.401824655257E-11,1.683376997149E+00,
19.817858914969E+80,1.506414967722E+29,4.394749884481E-01,
24.717851690435E-43,1.562793740651E-22,5.028946984109E+01/
DATA GOR/.010413309/
IF(H.GE.0.) GO TO 5
T=288.15-1.9812E-3*H
S=1.-2.9262913E-5*H
RETURN
5 IF(H.LE.750000.) GO TO 10
T=180.65
S=8.111816E-18
RETURN
10 CONTINUE
DO 20 I=1,8
20 IF(H.GE.HM(I))M=I
T=TM(M)+LM(M)*(H-HM(M))
IF(LM(M).EQ.0.)30,40
30 S=CS(M)*EXP(-GOR*H/T)
RETURN
40 S=CS(M)*T**(-1.-GOR/LM(M))
RETURN
END
```

C SUBROUTINE SLIN1(X, Y, AX, AY, N)
SINGLE LINEAR INTERPOLATION SUBROUTINE
DIMENSION AX(N), AY(N)
IF(N.EQ.1) GO TO 30
IF(X.LE.AX(1)) GO TO 20
IF(X.GT.AX(N)) GO TO 30
DO 10 I=1,N
K = I
Z = X - AX(I)
IF(Z.LE.0.) GO TO 11
10 CONTINUE
11 J=K-1
S = (AY(K) - AY(J))/(AX(K) - AX(J))
Y = AY(J) + S*(X - AX(J))
GO TO 100
20 Y=AY(1)
GO TO 100
30 Y=AY(N)
100 RETURN
END

```
SUBROUTINE SGX(N,X,M,NFEST)
DIMENSION X(16),XF(16),XB(16),CF(16),CB(16)
COMMON/VFO1D/G(50)
COMMON/VFO1F/GC(25,50)
COMMON/SS/SS(50)
COMMON/IGX/IGX
DATA EPSP/1.E-4/
IGX=1
DO 15 I=1,N
XF(I)=X(I)
15 XB(I)=X(I)
DO 20 I=1,N
DX=ABS(EPSP*X(I))
IF(ABS(X(I)).LE.EPSP) DX=EPSP**2
XF(I)=X(I)+DX
CALL SG(XF,FF)
DO 16 J=1,M
16 CF(J)=SS(J)
XF(I)=X(I)
XB(I)=X(I)-DX
CALL SG(XB,FB)
NFEST=NFEST+2
DO 17 J=1,M
17 CB(J)=SS(J)
XB(I)=X(I)
G(I)=0.5*(FF-FB)/DX
DO 20 J=1,M
GC(I,J)=.5*(CF(J)-CB(J))/DX
20 CONTINUE
IGX=0
RETURN
END
```

APPENDIX D
GLOSSARY OF VARIABLES IN USER CODE

PROGRAM MRRV

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
AK	Real	Measure of constraint convergence rate
AKMIN	Real	Defines convergence for constraints
C(I)	Real	Constraint residuals and constraint scale factors
DFN	Real	Expected decrease in performance index
DPR	Real	Degrees per radian
EPS(I)	Real	Defines convergence in VA09A
GC(I)	Real	Derivatives of the constraints
G2P(I)	Real	Second derivative matrix in factored form
I	Integer	Counter
IFN	Integer	Number of function evaluations in VA09A
INOM	Integer	Trajectory print flag. Print occurs if INOM = 1
IPROB	Integer	Problem flag. = 1, reentry; = 2, plane change
IPR1	Integer	Print flag in VF01A
IPR2	Integer	Print flag in VA09A
IRS	Integer	Restart flag. = 0, first run; = 1, restart
ITN	Integer	Number of iterations in VA09A
IW	Integer	Number of elements in work space W(I)
K	Integer	Number of equality constraints
M	Integer	Number of constraints
MAXFN	Integer	Maximum number of function evaluations in VA09A
MC	Integer	= M
ME	Integer	Number of equality constraints
MI	Integer	Number of inequality constraints
MINS	Integer	Number of iterations in VF01A

PROGRAM MRRV, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
MODE	Integer	Flag which indicates how θ , σ , and G2P are input
M2	Integer	= 2M
N	Integer	Number of parameters
NA	Integer	Number of parameters in angle of attack table
NFEST	Integer	Number of function evaluations
NM	Integer	Number of parameters in bank angle table
NN	Integer	Number of elements in G2F
T(I)	Real	Array where θ and σ are stored
TA(I)	Real	Node ordinates for angle of attack table
TF	Real	Final time
TM(I)	Real	Node ordinates for roll angle table
TMAX	Real	Internal time limit
TTA(I)	Real	Node abscissas for angle of attack table
TTM(I)	Real	Node abscissas for roll angle table
T0	Real	Initial value of internal time
T1	Real	Current value of internal time
X(I)	Real	Initial values of unknown parameters

SUBROUTINE VF01B

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
C(I)	Real	Constraints and constraint scale factors
F	Real	Performance index
GC(I)	Real	Derivatives of F with respect to X(I)
GC(I,J)	Real	Derivatives of C(J) with respect to X(I)
IGAMMA	Integer	Flag; = 0, $\gamma > -60$ deg ; = 1, $\gamma \leq -60$ deg

SUBROUTINE VF01B, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
IS	Integer	Used in optimization code
K	Integer	Number of equality constraints
M	Integer	Number of constraints
MM	Integer	Used in optimization code
MMK	Integer	Used in optimization code
N	Integer	Number of parameters
NFEST	Integer	Number of function evaluations
NU	Integer	Used in optimization code
X(I)	Real	Unknown parameters

SUBROUTINE SG

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
A	Real	Angle of attack (rad)
CF	Real	$\cos \phi$
CI	Real	$\cos i$
CSI	Real	$\cos \psi$
DELT	Real	Integration step size
DELT1	Real	Partial integration step before engine is started or shut off
DELT2	Real	Partial integration step after engine is started or shut off
DPR	Real	Degrees per radian
DY(I)	Real	Right-hand sides of the differential equations of motion
F	Real	Performance index

SUBROUTINE SG, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
I	Integer	Counter
IGAMMA	Integer	Flag; = 0, $\gamma > -60$ deg; = 1, $\gamma \geq -60$ deg
IGX	Integer	Flag; = 1, prevents update of constraint residuals when derivatives are being computed
INOM	Integer	Trajectory print flag; = 1, causes trajectory to be printed
IPROB	Integer	Problem flag; = 1, reentry; = 2, plane change
J	Integer	Counter
KOUNT	Integer	Engine flag; = 0, engine off; = 1, engine on; = 2, engine off again
M	Real	Bank angle (rad)
MASS	Real	Vehicle mass (nondimensional)
MC	Integer	Number of constraints
ME	Integer	Number of equality constraints
MI	Integer	Number of inequality constraints
N	Real	Load factor
NA	Integer	Number of parameters in angle of attack table
NDE	Integer	Number of differential equations
NIS	Integer	Number of integration steps
NM	Integer	Number of parameters in bank angle table
PA	Real	Angle of attack (deg)
PM	Real	Bank angle (deg)
S(I)	Real	Constraints ($\equiv C(I)$)
SS(I)	Real	Temporary values of constraints
SSI	Real	$\sin \psi$
T	Real	Time (normalized)

SUBROUTINE SG, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
TA(I)	Real	Node ordinates in angle of attack table (rad)
TEMP	Real	Temporary computation
TF	Real	Final time (nondimensional)
TIME	Real	Time (sec)
TIMEL	Real	Ignition time or burnout time (sec)
TIMETB	Real	Ignition time (nondimensional)
TIMETL	Real	Ignition time or burnout time (nondimensional)
TM(I)	Real	Node ordinates of bank angle table (rad)
TTA(I)	Real	Node abscissas of angle of attack table (nondimensional)
TTM(I)	Real	Node abscissas of bank angle table (nondimensional)
VCGOR	Real	$V \cos \gamma / r$ (nondimensional)
W	Real	Angular velocity of earth (nondimensional)
X(I)	Real	Unknown parameters
Y(I)	Real	State variables (nondimensional)
Z(I)	Real	State variables (dimensional)

SUBROUTINE RUNGE

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
DELT	Real	Stepsize
DELX(I,J)	Real	Increment in X(I)
DX(I)	Real	Derivative of X(I)
I	Integer	Counter
N	Integer	Number of differential equations

SUBROUTINE RUNGE, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
T	Real	Independent variable
T2	Real	Intermediate value of independent variable
X(I)	Real	Dependent variables
XV(I)	Real	Intermediate value of X(I)

SUBROUTINE DERIV

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
A	Real	Angle of attack (rad)
ALIM	Real	Angle of attack limit (rad)
CA	Real	$\cos \alpha$
CF	Real	$\cos \phi$
CG	Real	$\cos \gamma$
CM	Real	$\cos \mu$
CS	Real	$\cos \psi$
D	Real	Drag (nondimensional)
DY(I)	Real	Right-hand sides of the equations of motion
F	Real	Latitude, ϕ (rad)
FDOT	Real	$\dot{\phi}$ (nondimensional)
GR	Real	Acceleration of gravity (nondimensional)
KOUNT	Real	Engine flag; = 0, off; = 1, on; = 2, off again
L	Real	Lift (nondimensional)
M	Real	Bank angle (rad)

SUBROUTINE DERIV, Cont'd.

Variable	Type	Definition
MASS	Real	Mass (nondimensional)
N	Real	Load factor
NA	Integer	Number of nodes in angle of attack table
NDE	Integer	Number of differential equations
NLIM	Real	Load factor limit
NM	Integer	Number of nodes in bank angle table
P	Real	Temporary computation
R	Real	Distance from center of earth (nondimensional)
RDOT	Real	\dot{r} (nondimensional)
S	Real	Heading angle, ψ (rad)
SA	Real	$\sin \alpha$
SDOT	Real	$\dot{\psi}$ (nondimensional)
SP	Real	$\sin \phi$
SG	Real	$\sin \gamma$
SM	Real	$\sin \omega$
SS	Real	$\sin \psi$
T	Real	Longitude, θ (rad)
TA(I)	Real	Node ordinates in angle of attack table (rad)
TDOT	Real	$\dot{\theta}$ (nondimensional)
TF	Real	$\tan \phi$
TG	Real	$\tan \gamma$
THR	Real	Thrust (nondimensional)
TIMETB	Real	Ignition time (nondimensional)
TM(I)	Real	Node ordinates in bank angle table (rad)

SUBROUTINE DERIV, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
TPLOM	Real	$(T \sin \alpha + L)/m$
TMDOM	Real	$(T \cos \alpha - D)/m$
TT	Real	Normalized time, τ
TTA(I)	Real	Node abscissas in angle of attack table
TTF	Real	Final time (nondimensional)
TTM(I)	Real	Node abscissas in bank angle table
TW	Real	2ω (nondimensional)
V	Real	Velocity (nondimensional)
VCG	Real	$V \cos \gamma$
VCGOR	Real	$V \cos \gamma/r$
VDOT	Real	\dot{V} (nondimensional)
W2	Real	ω^2 (nondimensional)
Y(I)	Real	Nondimensional states

SUBROUTINE AERO

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
AL	Real	Angle of attack (rad)
ALFA	Real	Angle of attack (deg)
A1	Real	Coefficient in C_A_{SF}
A2	Real	Coefficient in C_A_{PR}
A3	Real	Coefficient in C_N
B1	Real	Coefficient in C_A_{SF}
B2	Real	Coefficient in C_A_{PR}
B3	Real	Coefficient in C_N
CA	Real	Axial force coefficient

SUBROUTINE AERO, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
CAL	Real	$\cos \alpha$
CAPR	Real	Axial force coefficient, pressure
CASF	Real	Axial force coefficient, skin friction
CD	Real	Drag coefficient
CL	Real	Lift coefficient
CN	Real	Normal force coefficient
C1	Real	Coefficient in $C_{A_{SF}}$
C2	Real	Coefficient in $C_{A_{PR}}$
C3	Real	Coefficient in C_N
D	Real	Drag (nondimensional)
DPR	Real	Degrees per radian
H	Real	Altitude (ft)
IPROB	Integer	Problem flag; = 1, reentry; = 2, plane change
L	Real	Lift (nondimensional)
M	Real	Mach number
R	Real	Distance from center of earth (nondimensional)
SAL	Real	$\sin \alpha$
SIGMA	Real	Density ratio
TAU	Real	Absolute temperature (deg K)
TA1(I)	Real	Table for A1
TA3(I)	Real	Table for A3
TB1(I)	Real	Table for B1
TB3(I)	Real	Table for B3
TC1(I)	Real	Table for C1

SUBROUTINE AERO, Cont'd.

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
TC3	Real	Table for C3
TM(I)	Real	Table for Mach number
T1A2(I)	Real	Table for A2, $\alpha \leq 16$ deg
T1B2(I)	Real	Table for B2, $\alpha \leq 16$ deg
T1C2(I)	Real	Table for C2, $\alpha \leq 16$ deg
T2A2(I)	Real	Table for A2, $\alpha > 16$ deg
T2B2(I)	Real	Table for B2, $\alpha > 16$ deg
T2C2(I)	Real	Table for C2, $\alpha > 16$ deg
VZL	Real	Velocity (ft/sec)
v	Real	Velocity (nondimensional)

SUBROUTINE ATM62

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
CS(I)	Real	Coefficients in density formula (nondimensional)
GOR	Real	g_S/R
H	Real	Altitude (ft)
HM(I)	Real	Altitude at beginning of layer m (ft)
I	Integer	Counter
LM	Real	Temperature gradient in layer m (deg K/ft)
M	Real	Denotes layer
S	Real	Density ratio
T	Real	Absolute temperature (deg K)
TM	Real	Temperature at beginning of each layer (deg K)

SUBROUTINE SLIN1

<u>Variable</u>	<u>Type</u>	<u>Definition</u>
AX(I)	Real	Node abscissas
AY(I)	Real	Node ordinates
I	Integer	Counter
J	Integer	Counter
K	Integer	Counter
N	Integer	Number of points in table
S	Real	Slope
X	Real	Abscissa
Y	Real	Ordinate
Z	Real	Temporary computation